Adaptive Sensors, Incorporated
216 Pico Blvd., Suite 8, Santa Monica, CA 90405 (213) 396 5997

13

ADAPTIVE CANCELLATION OF SCATTERED INTERFERENCE

L. E. Brennan, W. L. Doyle and I. S. Reed

March 1984

A Final Report

Submitted to

The Naval Air Systems Command

by

Adaptive Sensors, Inc.
216 Pico Boulevard, Suite 8
Santa Monica, CA 90405

on

Contract N00019-82-C-0488

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (Continued)

1. INTRODUCTION

This is the final report on a 1 year study of methods of adaptively cancelling scattered jamming. The results contained in this report are relevant to interference scattered into the main beam of a radar or communication system from terrain or chaff illuminated by a jammer. These same techniques can be applied to jamming scattered into the sidelobes or main beam of a receiving antenna from scatterers near the antenna, i.e., the multipath problem, which is an important limitation in some adaptive nulling systems.

The technique for cancelling scattered interference, utilizing delayed replicas of the jamming signal received by an auxiliary antenna, is outlined in Section 2. In some applications of the scatter canceller a large number of adaptive weights are required. This problem arises in airborne radars when a jammer is illuminating a large area of terrain or volume of chaff in the main beam. An efficient algorithm for weight computation is required in these cases and is described in Section 3. Motion of the receiving antenna or jammer also complicates the problem and leads to a requirement for rapid adaptive weight updating. A simulation program, discussed in Section 4, was written to investigate the performance of the scatter canceller using the simple algorithm for weight computation. This simulation includes the effects of motion of the receiving antenna or jammer. Results of the simulation are contained in Sections 4 and 5 of this report.

The conclusions of the study are contained in Section 6. Results of the simulation show that the simple weight updating algorithm provides adaptive weights which converge quickly to near-optimum values. In most cases of interest, the adaptive weights provide good scatter cancellation when the receiving antenna or jammer are moving at typical aircraft velocities.

2. CANCELLATION OF SCATTERED INTERFERENCE

Many future radar and communication systems will employ directional receiving antennas with low sidelobes, plus adaptive systems for nulling jamming received directly through the sidelobes. In these systems with low vulnerability to direct sidelobe jamming, a second type of jamming may limit performance, viz., jamming scattered into the main beam from terrain or chaff. The problem is illustrated in Fig. 1, where the jamming is scattered from chaff in the main beam. A similar problem arises in airborne radars when airborne jammers are illuminating terrain in the main beam. Angular nulling cannot be used to reject this type of interference, since both the scattering medium and desired signal source (or target) are in the main beam. The scattered jamming consists of a large number of delayed replicas of the signal radiated by a jammer.

A similar problem arises in some systems due to scattering from objects near the receiving antenna. A distributed array of scatterers may reflect a set of delayed replicas of jamming into the antenna, either through the sidelobes or main beam. This multipath effect is an important limitation on the performance of sidelobe cancellers in some cases. When the scatterer locations and system bandwidth are such that the differential delays are the order of a reciprocal bandwidth or greater, the technique discussed below can be used to improve canceller performance.

A method of cancelling the scattered jamming has been described in earlier progress reports on this contract, and is illustrated in Fig. 2. The jamming signal is scattered into the main beam with a total delay of $(R_1+R_2)/c$, where c is the velocity of propagation. This delay

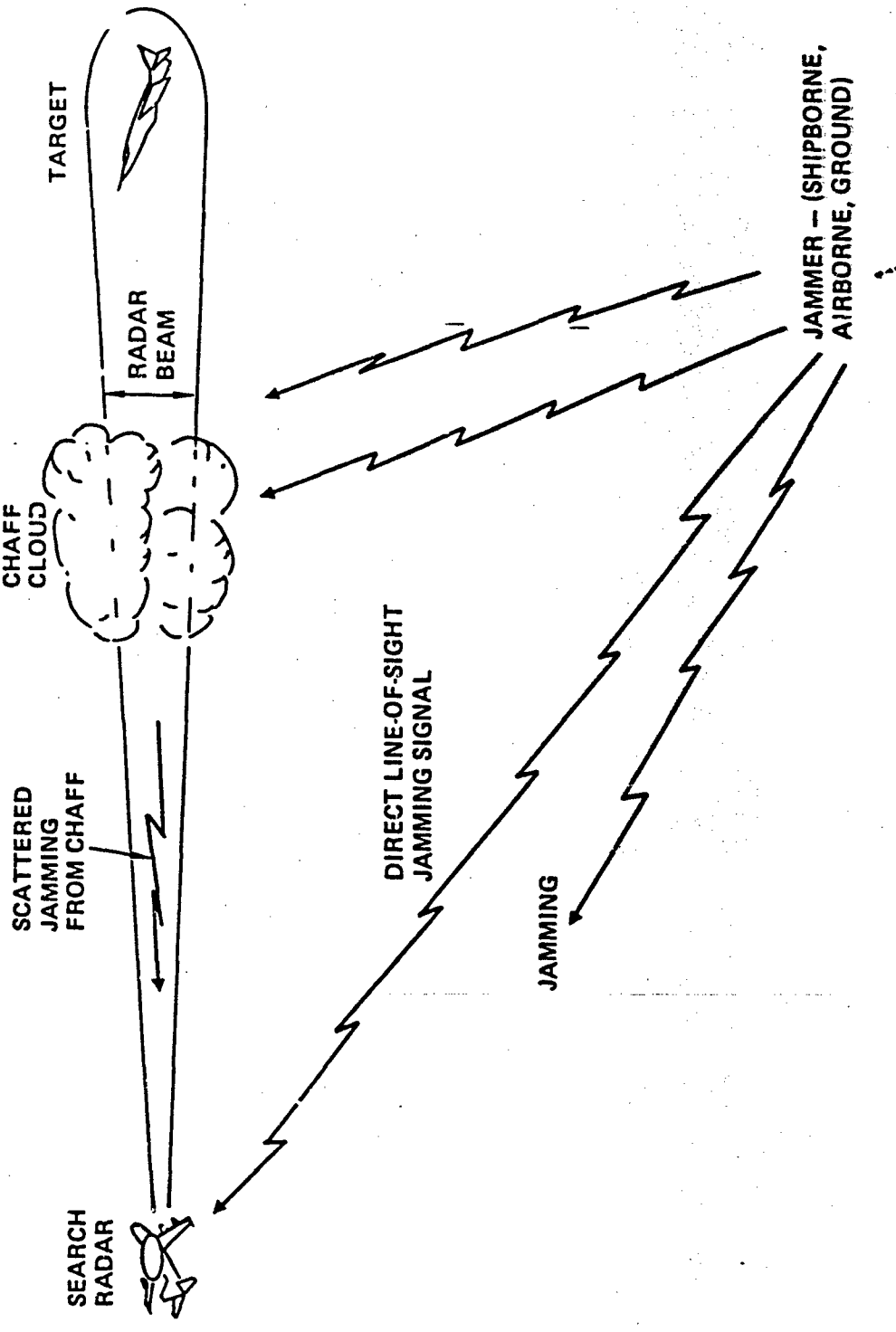

Fig. 1 -- Scattered Jamming into Main Receiving Beam

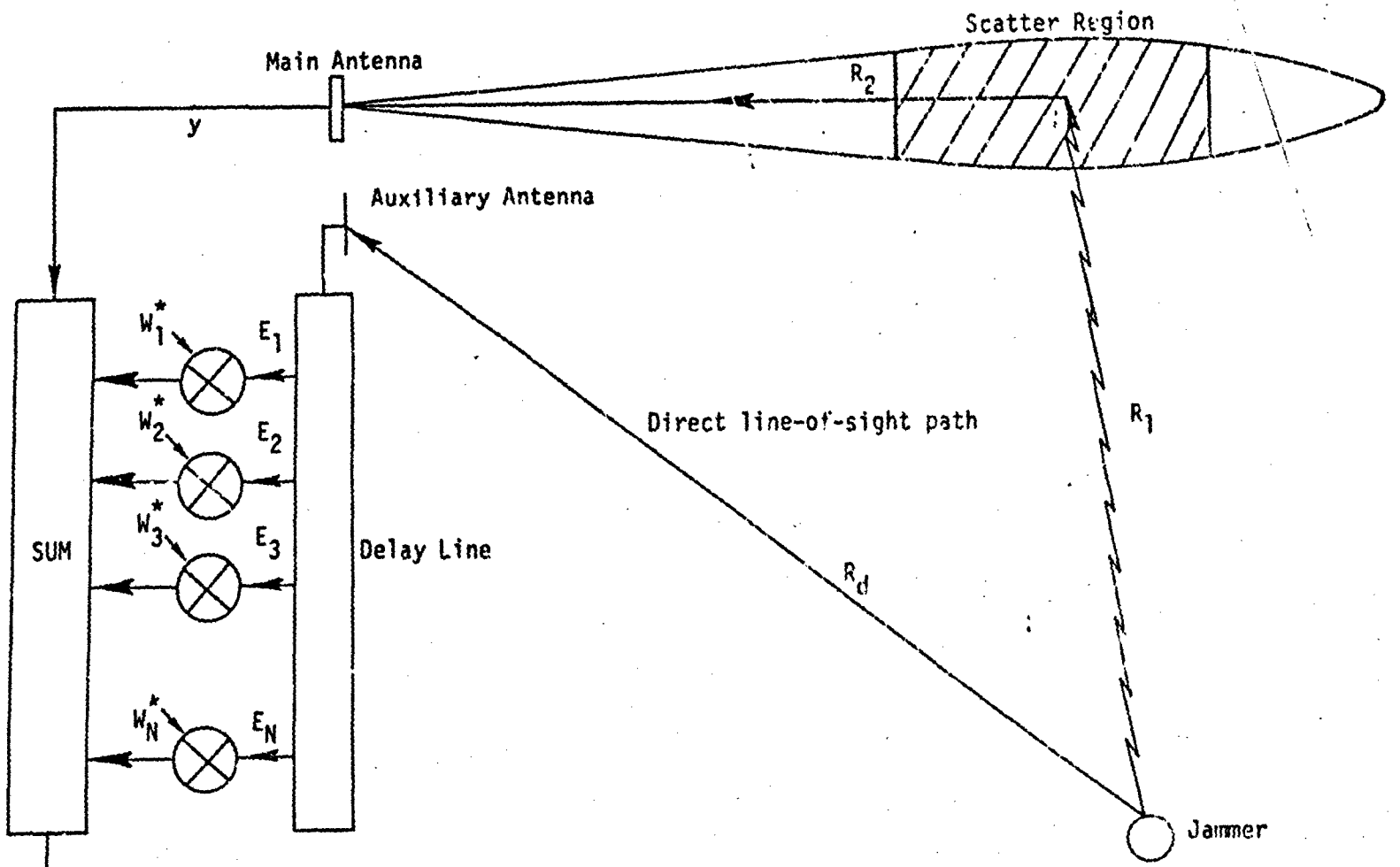

Fig. 2 -- Scatter Cancellation

extends over a time interval which depends on the geometry and the extent of the scattering medium. A replica of the jamming signal, delayed by R_d/c, is observed by a second auxiliary antenna located near the main receiving antenna. A set of delayed replicas of this auxiliary signal is adaptively weighted and subtracted from the main antenna output. The spacing between taps is roughly the reciprocal of the jammer (or receiver) bandwidth and the taps cover a delay interval corresponding to the extent of $(R_1+R_2-R_d)/c$. When the jamming is scattered into the main beam from chaff or terrain which is far from the receiving antenna, an omnidirectional auxiliary antenna can be used to obtain the jammer replica for scatter cancellation. The output of an omnidirectional auxiliary antenna will generally have a direct line-of-sight jammer signal large compared to the jammer scattered into the omnidirectional antenna. This ratio of direct to scattered jamming components in the auxiliary antenna must be large to achieve good scatter cancellation. When the scattering is via multipath from objects near the antenna, the auxiliary antenna pattern must be selected to assure that this ratio is large.

The output of the scatter canceller, Fig. 2, is

$$Z = Y - W^*E, \tag{1}$$

where W^* denotes a row vector of the adaptive weights, E is a column vector of the top outputs E_n, and * denotes the conjugate transpose. The jammer residue in the output is minimized when

$$W = M^{-1} S, \tag{2}$$

where M is the covariance of tap outputs $\overline{E\,E^*}$ and S is the column vector $\overline{E\,y^*}$. The scatter canceller is analogous to a multi-channel coherent sidelobe canceller, where the tap outputs in Fig. 2 correspond to auxiliary element outputs in a sidelobe canceller.

While a canceller for scattered jamming could be implemented with an analog delay line as shown in Fig. 2, it is more likely that future systems will use a digital implementation. Note that a radar or communication system using digital sidelobe cancellation for nulling sidelobe jamming has the necessary digital data available for main beam scatter cancellation. The digital data from the main beam and auxiliary channel can be combined as in Fig. 2 to implement scatter cancellation. Any of the various algorithms used in sidelobe cancellers for weight computation can be used in the scatter canceller.

3. COMPUTATION OF ADAPTIVE WEIGHTS

When jamming is scattered into the main beam from a large area or volume of scatterers, a large number of taps and adaptive weights may be required. For example, in a system with a 1 MHz bandwidth, the spacing between consecutive taps or samples must be no greater than 1 microsecond. If the scattering region extends over a 100 mile interval in delay, 600 or more adaptive weights are required.

A straightforward method of updating the weights is the estimation of M and S in Eq. (2) from sample second moments of the received signals. The weights can then be computed from Eq. (2) by solving a system of N simultaneous linear equations, where N is the number of weights. For N the order of 600, this is a formidable computation. As discussed in earlier reports, the true covariance matrix is Toeplitz, i.e., its elements M_{mn} are the same along each diagonal where (m-n) is constant. Since this matrix is also Hermitian, it is specified by only N numbers, e.g., the elements of the first row of the covariance matrix. For a Toeplitz matrix, the solution of Eq. (2) requires the order of N^2 multiplications, while for a general NxN matrix the order of N^3 multiplications are required. While the true covariance matrix of the tap outputs is strictly Toeplitz, a sample covariance matrix is only approximately Toeplitz. It is not known whether this Toeplitz assumption can be used to obtain sufficiently accurate weights, or how many samples must be included in the sample matrix in order to achieve good performance under this assumption.

In either case, with the order of N^3 or N^2 multiplications per weight update, the weight computation problem is formidable when rapid

updating and 600 or more weights are required. Thus it is important to find a simple algorithm for weight updating which minimizes the amount of computation. One of the simplest algorithms for computing digital adaptive weights is an iterative technique analogous to a multi-channel sidelobe canceller. Let E_i denote a column vector of consecutive digital samples, corresponding to the tap outputs in Fig. 2 at the i^{th} iteration. Also, let Z_i denote the i^{th} sample of the canceller output and W_i a column vector of the digital adaptive weights. The weight updating algorithm is

$$W_{i+1} = W_i - \gamma Z_i E_i^* \tag{3}$$

where γ is a constant. Both the convergence rate and weight jitter increase as γ is increased. Note that only N complex multiplies (plus scaling by the factor γ which can be a small constant, 2^{-m}) are required per iteration to update the adaptive weights. An additional N complex multiplies per input sample are required to form the output of Eq. (1). The weights can be updated on every sample or every n^{th} sample depending on convergence rate requirements.

In the simulation, the weights are initially set to zero. The program is written to simulate either a digital or an analog implementation. In the analog case, the weight iteration equation is

$$W_{i+1} = \alpha W_i - g(1-\alpha) Z_i E_i^* \tag{4}$$

where

$$\alpha = e^{-1/\tau}$$

τ = time constant

g = gain

The parameters g and τ are inputs to the program. For large τ, $\alpha \doteq 1$ and γ of Eq. (3) is g/τ. Large values of τ, simulating a digital system, were used in all of the simulation runs.

4. SIMULATION OF THE SCATTERED INTERFERENCE AND CANCELLER

The FORTRAN program for simulating scattered jamming and the scatter canceller includes the following steps: simulation of the jamming signal, specification of the field of scatterers, computation of the inputs to the main beam and auxiliary channel, simulation of the adaptive weight computer, evaluation of the canceller performance by plotting cancellation ratio as a function of time, computation of optimum weights and resulting cancellation ratio, and optionally, simulation of doppler offset due to motion.

The jammer signal in the simulation is represented by a series of complex Gaussian samples with independent zero-mean quadrature components of equal variance. The sample spacing is less than the reciprocal bandwidth by a factor I_s which is input to the program. A cosine frequency spectrum is simulated in all of the examples contained in this report. The method of generating the jammer signal is discussed in detail in Appendix B.

The scattering media of interest contain a large number of individual scatterers per interval of 1/B in $(R_1 + R_2 - R_d)/c$. Examples of scattering media are chaff a- diffusely scattering terrain. The scattering process is modeled as several scatterers per delay interval of 1/B, where the input parameter, I_s, to the program specifies this number. When there are a large number of scatters of comparable magnitude per delay interval of $1/(B\ I_s)$, each scatterer representing one of these delay intervals has a zero-mean Gaussian distribution in each quadrature component (from the

Central Limit Theorem). The computer program presently includes two options: complex Gaussian scattering coefficients selected randomly, and unit amplitude scatterers with uniformly distributed random phases.

The input to the auxiliary antenna via direct line of sight from the jammer is obtained from the sequence of jammer samples. At one time, the tap outputs are a set of these samples spaced I_s samples apart. The main beam input is obtained by multiplying the individual jammer samples by the corresponding scatterer coefficients and summing. Let $\{v_n\}$ denote a set of consecutive jammer samples, N_s the number of scatterers, and α_j the complex scatterer coefficients. The main beam inputs are $\{y_m\}$, where

$$y_m = \sum_{j=1}^{N_s} v_{(I_s m + j)} \alpha_j . \tag{5}$$

The corresponding tap outputs from the auxiliary channel, on the m^{th} sample for the n^{th} tap, are

$$E_n(m) = V_{[n_o + I_s(m-n)]} , \quad n = 1,2,... \tag{6}$$

where n_o is selected so that the tap samples bracket the scattering interval.

The canceller weights are set to zero at the beginning of each simulation run and iterated as in Eq. (4) of the preceding section. The measure of performance used in the program is the cancellation ratio, i.e., the ratio of scattered jamming residue after cancellation to the power in the main beam output before cancellation. This cancellation ratio is

is plotted as a function of time to show the transient response of the canceller. Each point on the curve is obtained by averaging the random output powers over IAVE contiguous samples. When this parameter is reduced, the fluctuations of the output are greater due to the random variation of the output from sample to sample. With IAVE=1, the plot shows the raw output of the canceller, which would be the input to the rest of the system.

The optimum weights and corresponding cancellation ratio depend on the covariance matrix of the tap outputs and the vector of cross correlations between the main channel output and tap outputs. A sample covariance matrix and correlation vector are computed for the input data samples used in each simulation run. The optimum weights and cancellation ratio based on these sample second moments are computed and printed for each run. With a large number of input samples, these sample matrix estimates are expected to closely approximate the corresponding theoretical weights and cancellation ratio. The cancellation ratio based on a sample of inputs and tested against these same samples will be slightly better than the theoretically optimum ratio based on the true covariance matrix. Both the sample matrix tap weights and the weights at the end of each simulation run are printed in the output. The sample matrix cancellation ratio is shown in each output curve. This ratio is relevant for comparison with the simulation curve of cancellation ratio versus time in the zero doppler (no motion) cases.

The results of one simulation run are shown in Fig. 3. In this example, the adaptive weights are iterated 480 times. The input samples

are spaced by 1/B, where B is the full zero-to-zero width of the cosine spectrum of the jamming signal. Since this spectrum is strictly band-limited and complex samples are obtained at intervals of 1/B, there is no limitation due to spectral aliasing. Thus, there is no limit to the cancellation which can be achieved as the number of optimized tap weights increases, since receiver noise is not included in the simulation. In the example, 16 taps are used, centered on an interval of 10 tap spacings which contain the scattering medium. There are 41 Gaussian scatterers distributed evenly over the 10 intervals of 1/B in delay. The plots are normalized so that the cancellation is zero dB at the beginning of each run. Each point on the curve is obtained by averaging the output power over 30 samples, i.e., IAVE=30 in this example. The ratio of τ/G is 50 in this example and the weights are iterated in accordance with Eq. (4). The value of τ for this example was 1000, so the simulation closely approximates a digital system with a γ of 1/50 in Eq. (3). Note that the adaptive weights converge quickly to near-optimum values. The output residue of scattered jamming is reduced by 30 dB in about 350 iterations. In a system with 1 MHz bandwidth, this corresponds to a convergence time of 0.35 milliseconds.

Since these results are obtained by simulation of the jamming signal, the output residue and adaptive weights depend on the random samples of jamming. The random number seed used in generating the jamming is shown in Figs. 3 and 4. The seed of the random number generator is different in the two examples, which are otherwise identical. Note that the detailed cancellation ratio vs. time curves are different

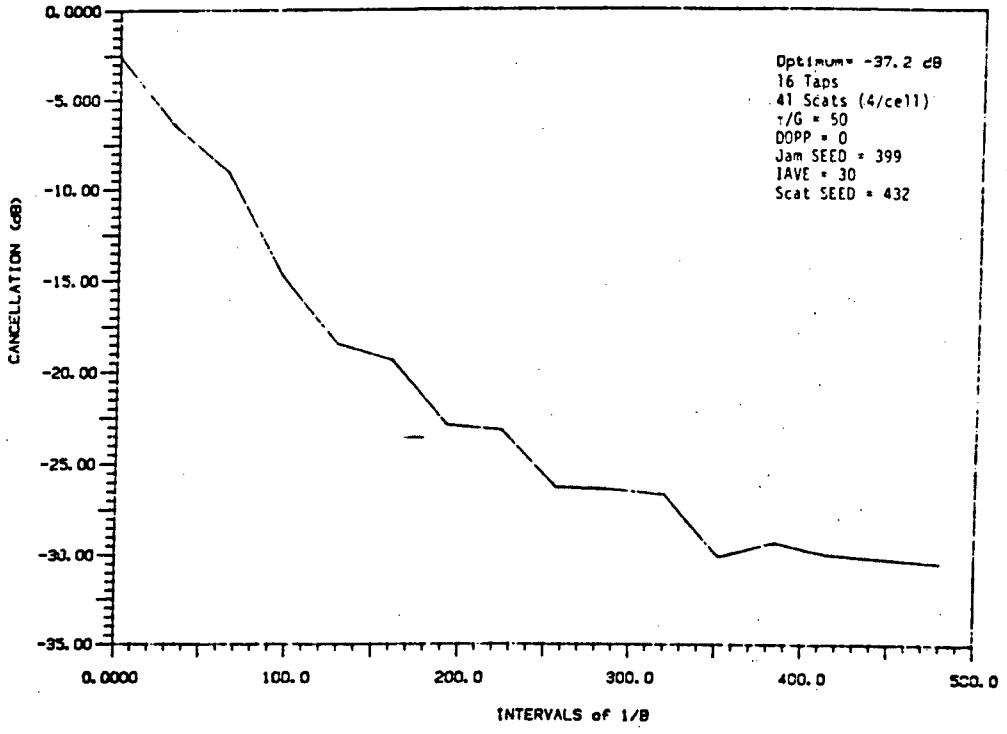

Fig. 3 -- Transient Response of Scatter Canceller

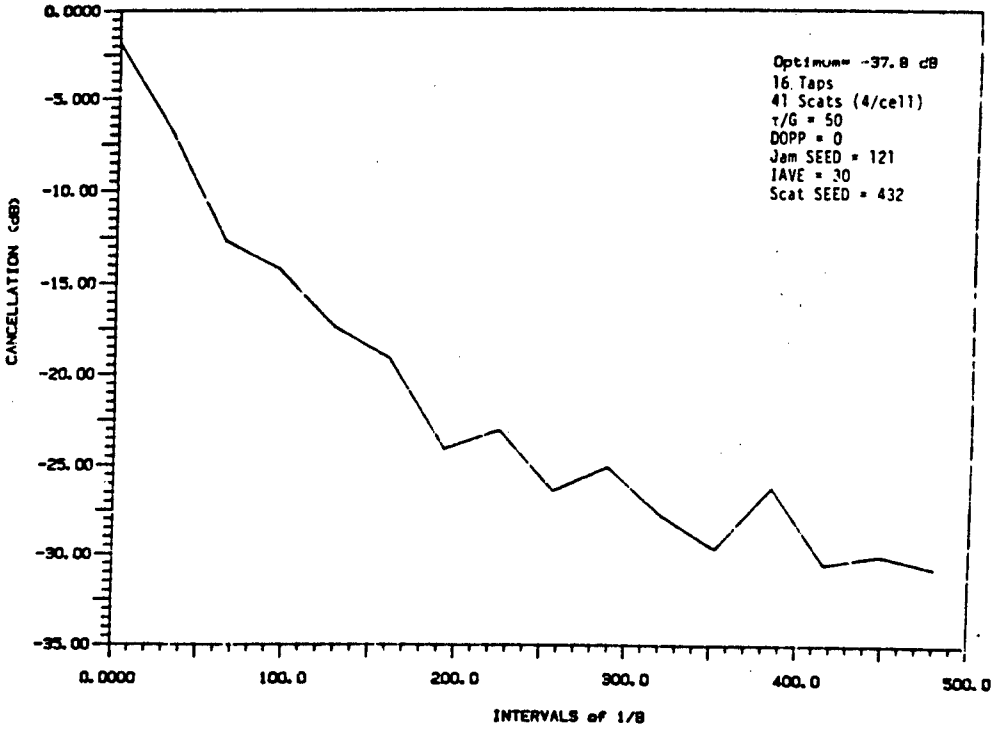

Fig. 4 -- Scatter Canceller - Effect of Random Jamming Samples
(Compare Fig. 3)

in the two cases. However, the convergence rates and final cancellation ratios are approximately the same, viz., 30 dB of cancellation after 350 iterations. The cancellation ratio achieved with optimum weights is shown in each case, 37.2 dB in Fig. 3 and 37.8 dB in Fig. 4. These ratios are obtained from a sample covariance matrix of the input random samples.

The complex scatterer coefficients are also generated randomly in the simulation. All parameters are the same in Figs. 4 and 5 except the random number seeds used in generating the scatterer coefficients. Note that these two curves differ in detail. Again, the results are similar and roughly 30 dB of cancellation is obtained after 450 iterations.

The next series of 5 examples, Figs. 6 through 10, shows the effects of the number of delay taps and adaptive weights on the cancellation ratio. In each of these cases, 10 delay cells of width 1/B contain scatterers. There are 41 scatterers evenly distributed over the 10 cells, and the scatterer amplitudes and phases are identical in the 5 examples. In Fig. 6 there are 10 taps which exactly cover the scattering region, but no taps outside this region. The cancellation ratio computed from the sample covariance matrix is only 16.9 dB in Fig. 6. The cancellation converges quickly to this level in the simulation. In Fig. 7, with 12 delay taps centered on the scattering region, the cancellation ratio is ~27 dB. The addition of one extra tap on each side of the scatterer delay interval increases the cancellation by about 10 dB. The next 3 examples, with 16, 24 and 34 taps respectively centered on the scattering interval, show the effect of tap number on cancellation. The cancellation improves with increasing number of extra taps. The additional taps with adaptive weights improve the ability of the system to interpolate between

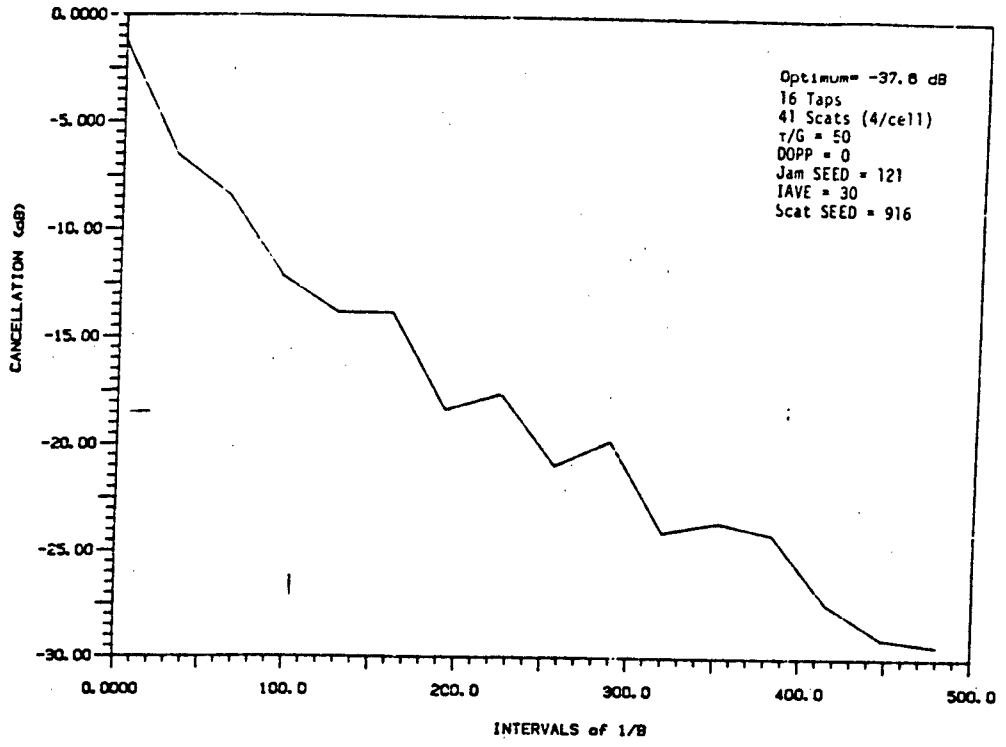

Fig. 5 -- Scatterer Canceller - Effect of Random Scatterer Coefficients (Compare Fig. 4)

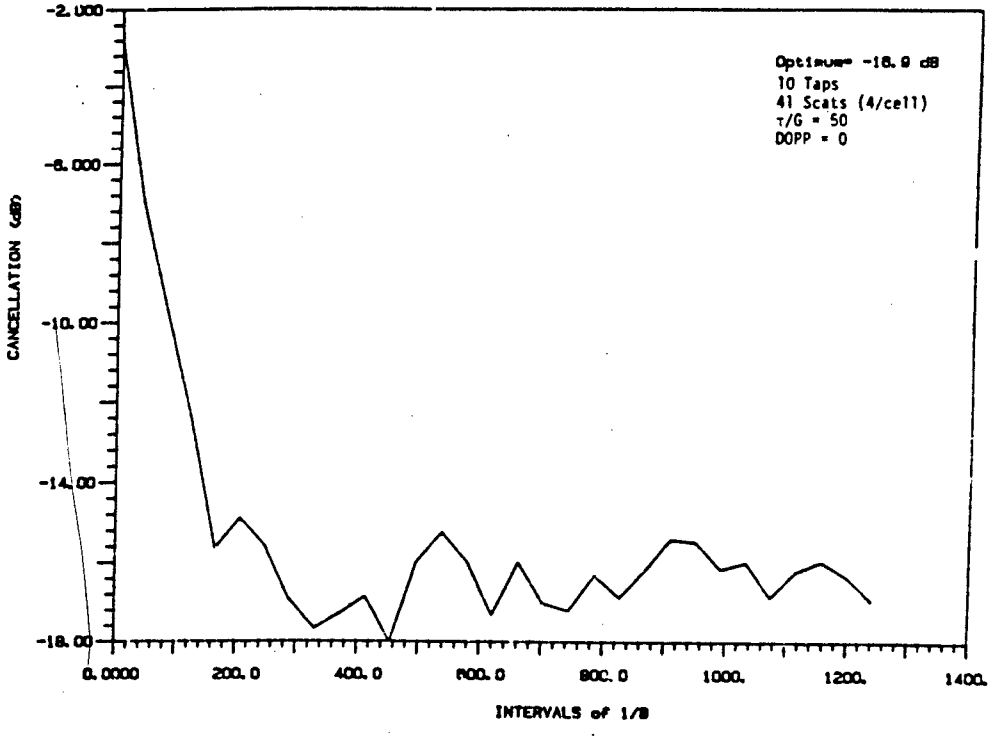

Fig. 6 -- Scatterer Canceller - 10 Taps, 10 Cells Contain Scatterers

tap samples and match the jamming waveforms reflected from scatterers between tap points. Convergence to near optimum weights is somewhat slower with a larger number of taps. This series of examples shows that, with a cosine frequency spectrum, the addition of 3 extra taps on each side of the scatter delay interval improves the cancellation ratio significantly. In Fig. 8, with 16 taps, the cancellation is ~38 dB compared with ~17 dB in Fig. 6 with 10 taps. Further increase in the numbe. of extra taps is useful but yields less improvement.

In the simulation, the distributed scattering medium is represented by a few random scatterers in each delay cell of width 1/B. The next series of 5 examples in Figs. 11 through 15 shows the effect of changing the number of scatterers per cell on performance. In each of these examples there are 12 delay taps spaced by 1/B. The τ/G ratio is 50 and the scatterers are distributed uniformly over 6 delay intervals of 1/B in each case. In Fig. 11 there are 2 scatterers per cell with one of the scatterers in each cell coincident in delay with an adaptively weighted tap. Somewhat better performance would be expected in this case where the scattered jamming signal from half of the scatterers can be cancelled exactly. The optimum cancellation ratio based on the sample covariance matrix is 33.9 dB in Fig. 11. The simulated performance approaches this level of cancellation and some points on the curve show greater than optimum cancellation. This is due to the random variation in noise samples, where only 10 output noise samples are averaged to obtain each point on the curve of Fig. 11. In Figs. 12 through 15, the number of scatterers per cell are 3, 4, 5, and 6, respectively. With 3 and 4 scatterers per cell the optimum cancellation ratio is ~28 dB. A smaller

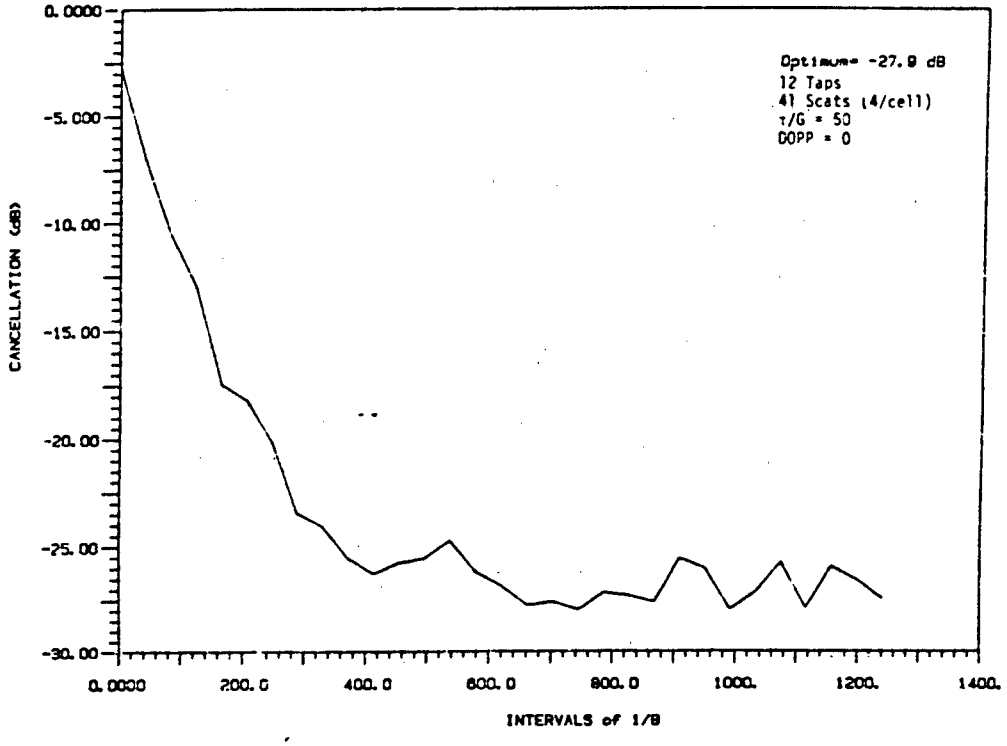

Fig. 7 -- Scatter Canceller - 12 Taps, 10 Cells Contain Scatterers

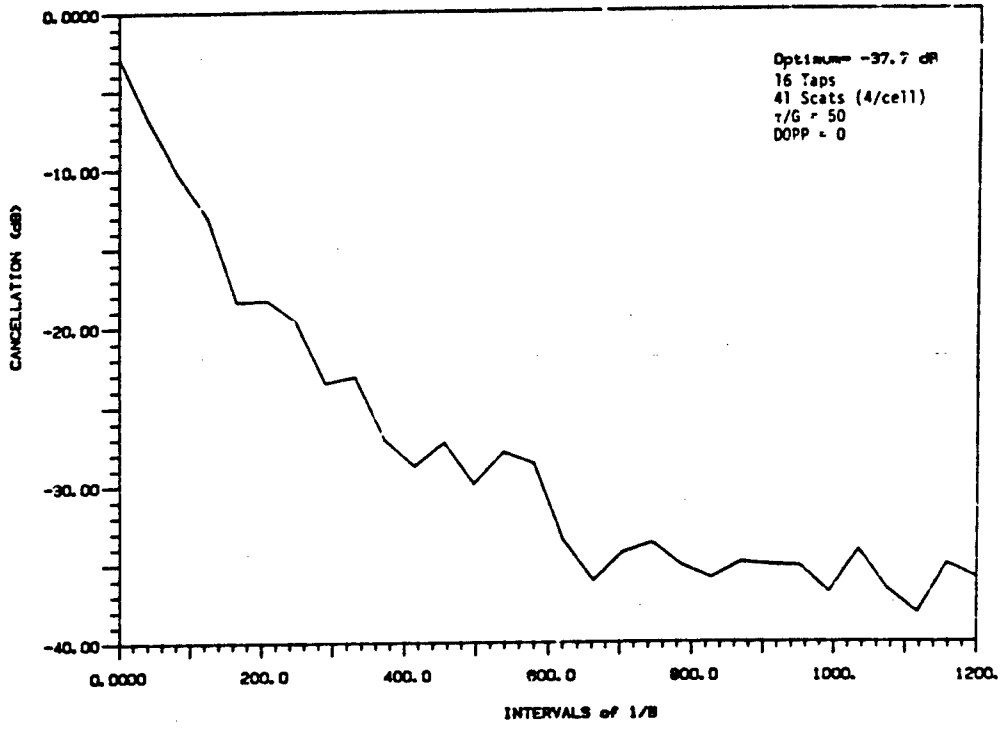

Fig. 8 -- Scatter Canceller - 16 Taps, 10 Cells Contain Scatterers

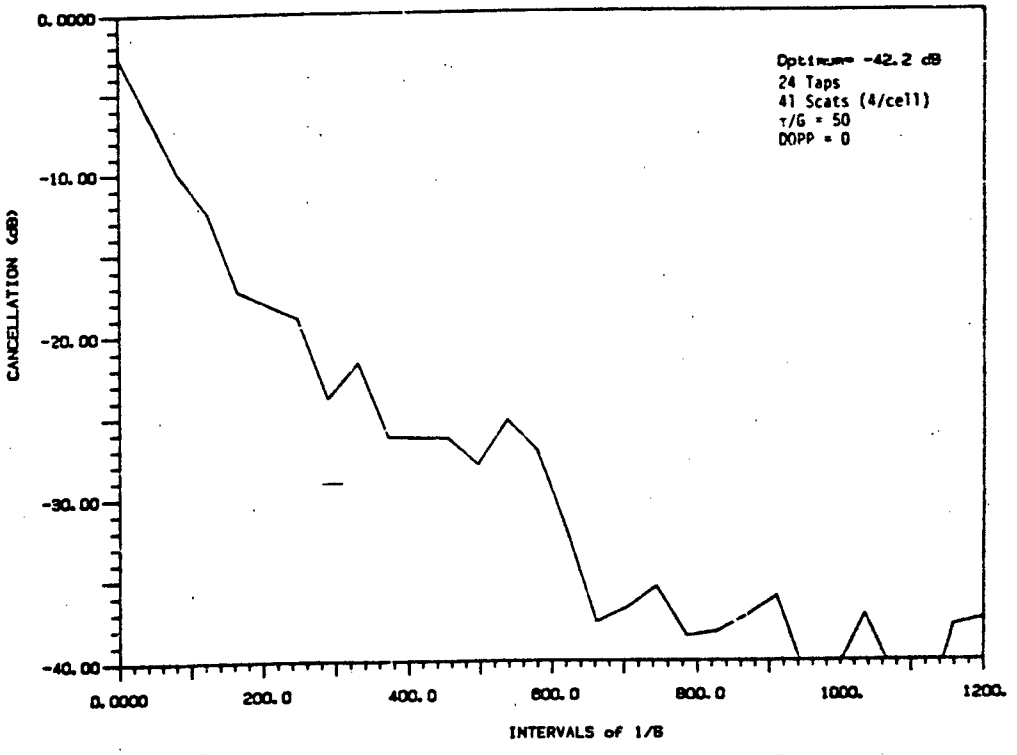

Fig. 9 -- Scatter Canceller - 24 Taps, 10 Cells Contain Scatterers

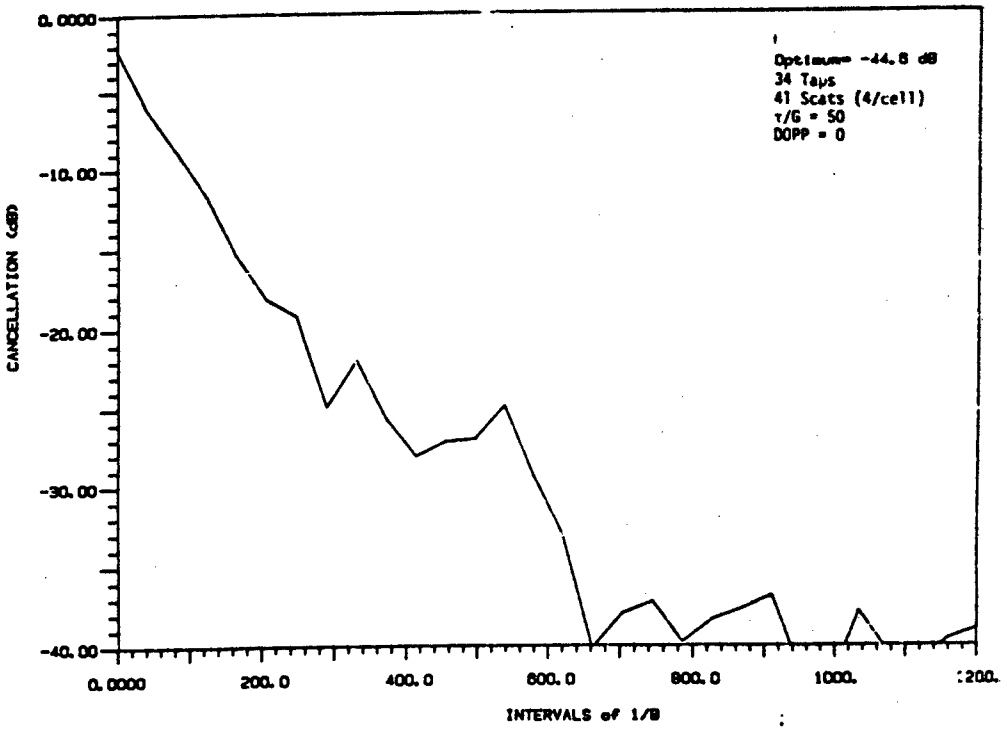

Fig. 10 -- Scatter Canceller - 34 Taps, 10 Cells Contain Scatterers

fraction of the scattered components are coincident with tap samples and can be cancelled exactly than in Fig. 11. With 5 and 6 scatterers per cell, in Figs. 14 and 15, the optimum cancellation ratio based on a sample covariance matrix of all the input samples increases to 32 and 33 dB, respectively. One would expect less cancellation as the number of scatterers per cell increases. This increase is small, however, and is probably due to the random variations in the jamming signal and scattering coefficients discussed earlier. From this series of examples, it appears that 4 scatterers/cell are sufficient to simulate distributed scattering and 4/cell are used in the remaining simulations.

The simulation program includes the option of unit amplitude scatters of random phase as well as the Rayleigh amplitude scatterers (Gaussian quadrature components). One example using the unit amplitude scatterers is shown in Fig. 16. In all other cases the Rayleigh scatterers are used. Comparing Figs. 13 and 16, with Rayleigh and unit amplitude scatterers, respectively, shows that the results are essentially the same in the two cases.

The next series of 3 examples in Figs. 17 through 19 shows the effect of varying the tap spacing. In each case there are 49 scatterers uniformly distributed over 12 intervals of 1/B in delay. In the reference case of Fig. 17 there are 18 taps centered on the scatterer delay interval and spaced by 1/B. In Fig. 18, there are 24 taps spaced by 3/4B and covering the same delay interval of 18 intervals of 1/B. Oversampling and increasing the number of adaptive weights in Fig. 18 does not significantly change the performance of the scatter canceller. This

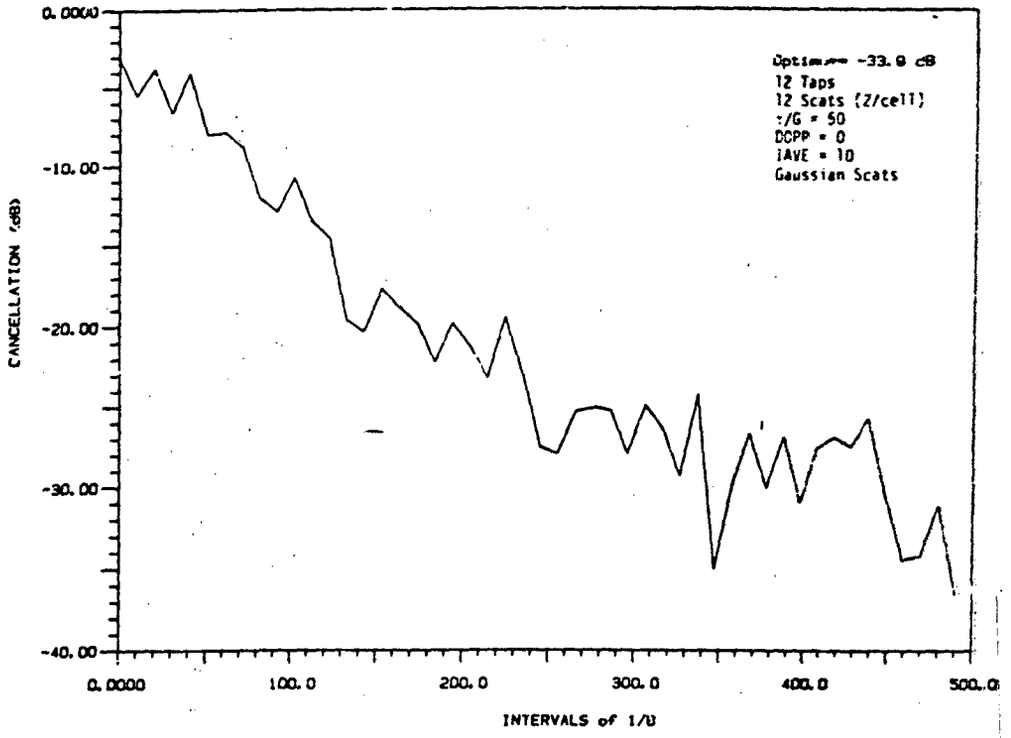

Fig. 11 -- Simulation of Scatter Canceller with 2 Scatterers per Cell

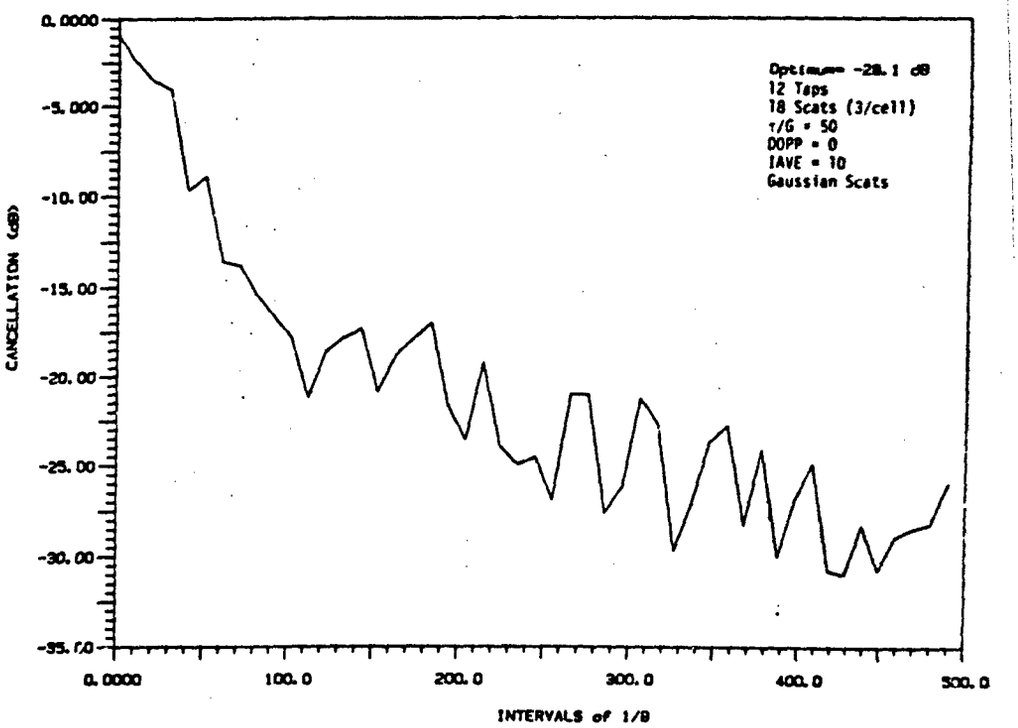

Fig. 12 -- Simulation of Scatter Canceller with 3 Scatterers per Cell

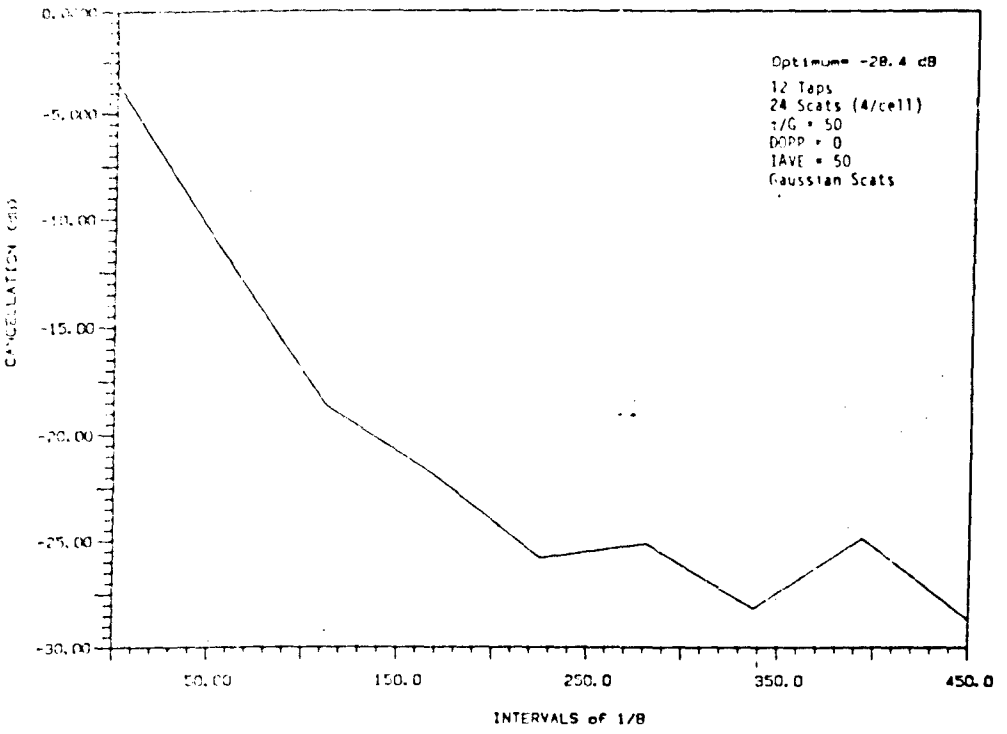

Fig. 13 -- Simulation of Scatter Canceller with 4 Scatterers per Cell

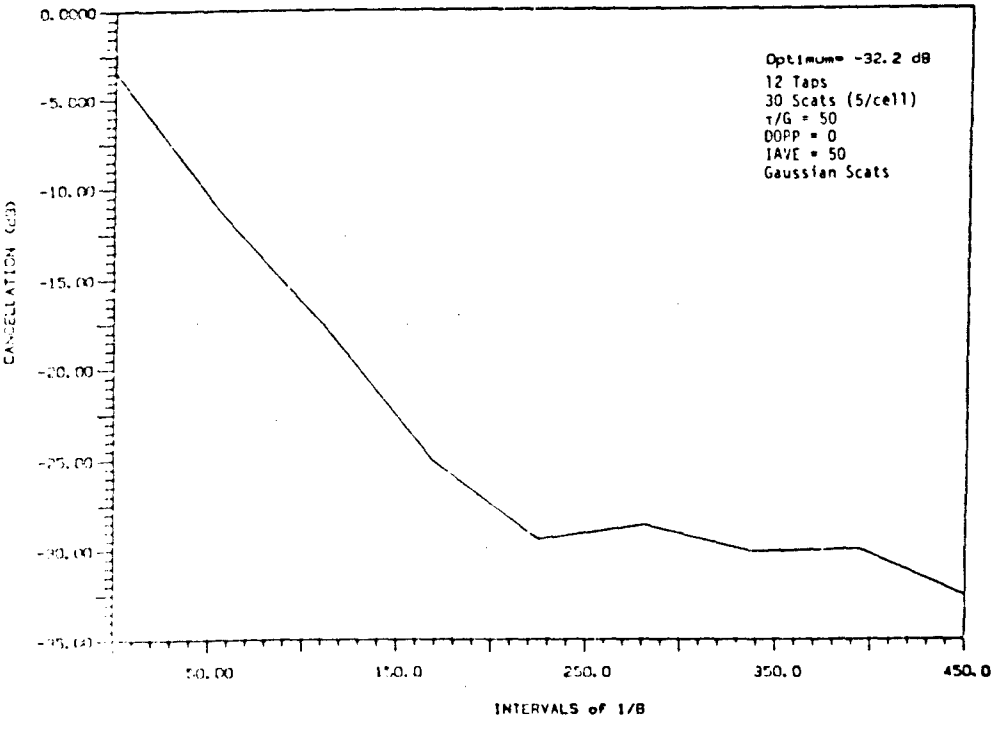

Fig. 14 -- Simulation of Scatter Canceller with 5 Scatterers per Cell

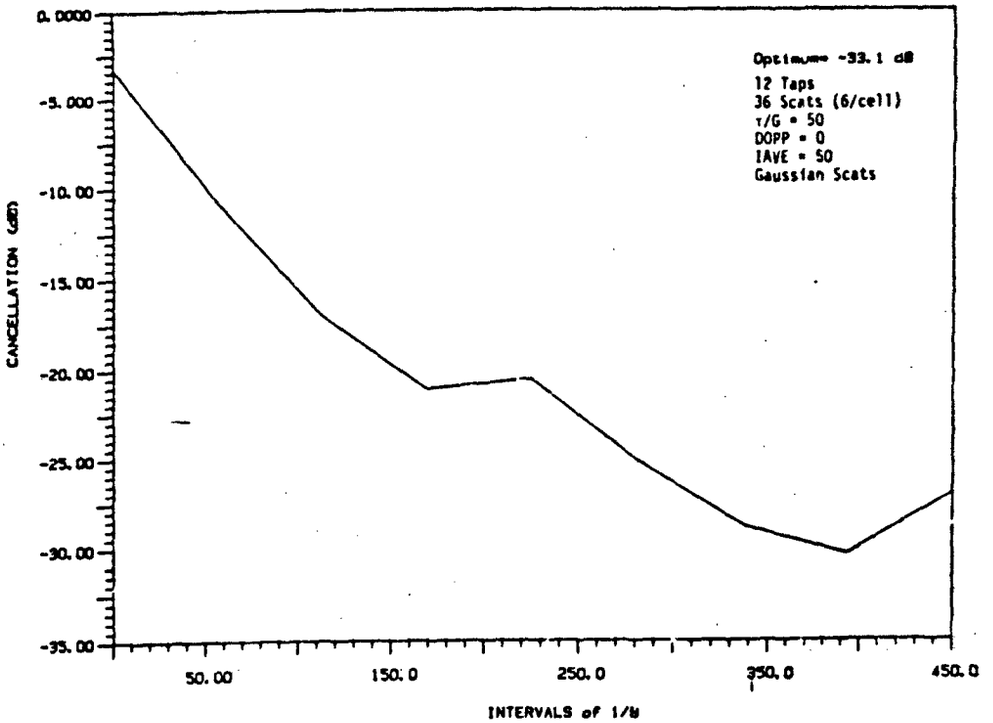

Fig. 15 -- Simulation of Scatter Canceller with 6 Scatterers per Cell

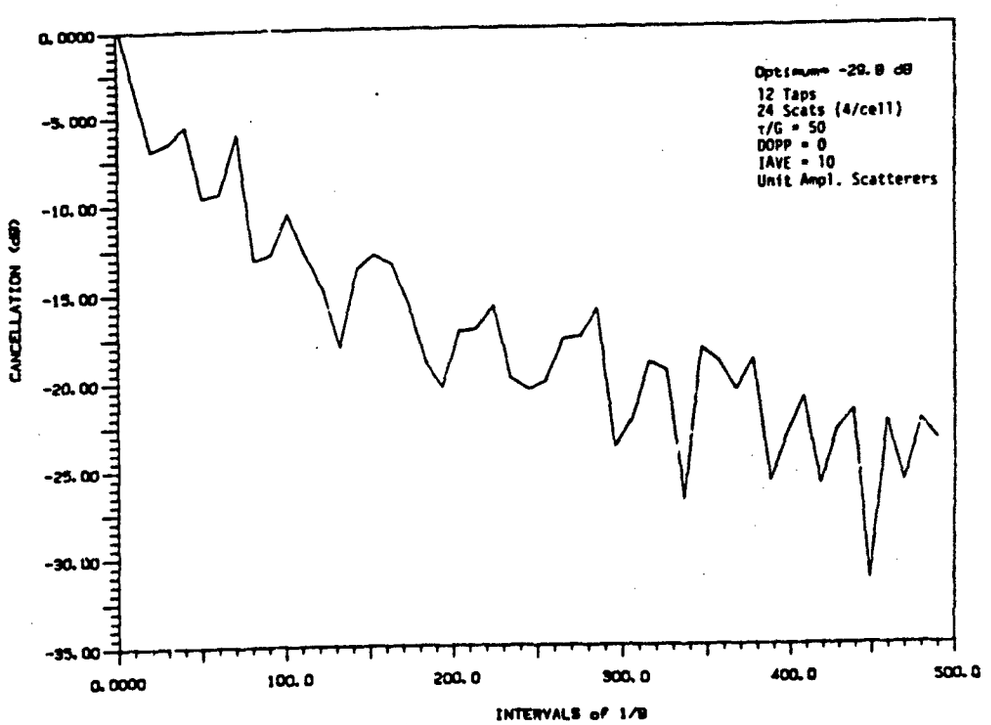

Fig. 16 -- Simulation of Scatter Canceller with Unit Amplitude Scatterers (4/cell)

is to be expected since the jammer signal is strictly bandlimited to a bandwidth B, so samples spaced by 1/B are sufficient for interpolation. In Fig. 19, the sample spacing is increased to 5/4B and the 14 delay taps cover approximately the same delay interval. The cancellation ratio is reduced drastically, to ~7 dB, in this case. As expected, when the sample spacing exceeds 1/B it is not possible to interpolate with sufficient accuracy to replicate the jamming signals from scatterers which are not colocated with tap samples. The sample spacing is 1/B in all of the simulation runs except Figs. 17 and 19.

The next three examples illustrate the effect of the time constant used in updating the adaptive weights, i.e., τ/G of Eq. (4) or γ^{-1} of Eq. (3). In each case there are 16 taps centered on 10 scattering intervals of width 1/B. In Fig. 20, the τ/G ratio is 10 and the adaptive weight fluctuations are large. The cancellation ratio drops below 0 dB at 1000 iterations. With a τ/G ratio of 25 in Fig. 21, the cancellation ratio converges to nearly the optimum value of 37.7 dB after 300 iterations and the fluctuations in output jammer residue are small. In Fig. 22, with a τ/G ratio of 100, convergence is slower. Approximately 1000 iterations are required to achieve near optimum cancellation.

The preceding examples show that the simple weight updating algorithm (Eqs. (3) or (4)) yields rapid convercence of the adaptive weights and cancellation of scattered jamming which approaches the cancellation with optimum weights. It is not necessary to implement a slower and/or much more costly weight computer utilizing the sample covariance matrix for weight updating. These results were obtained with

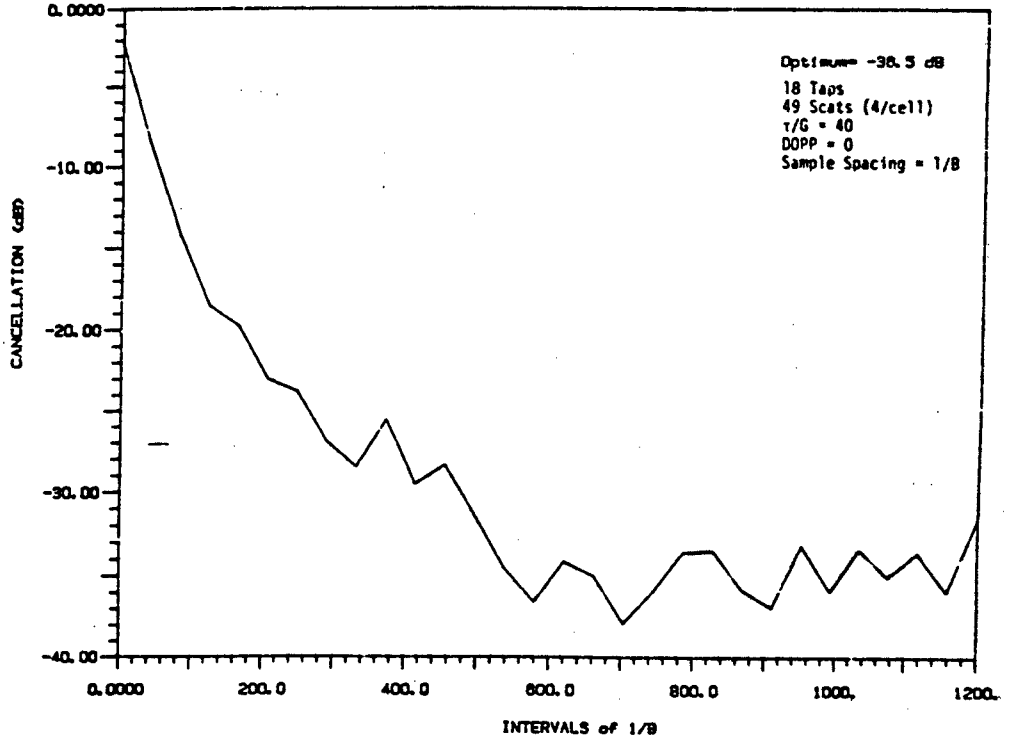

Fig. 17 -- Scatter Canceller - Samples Spaced by 1/B

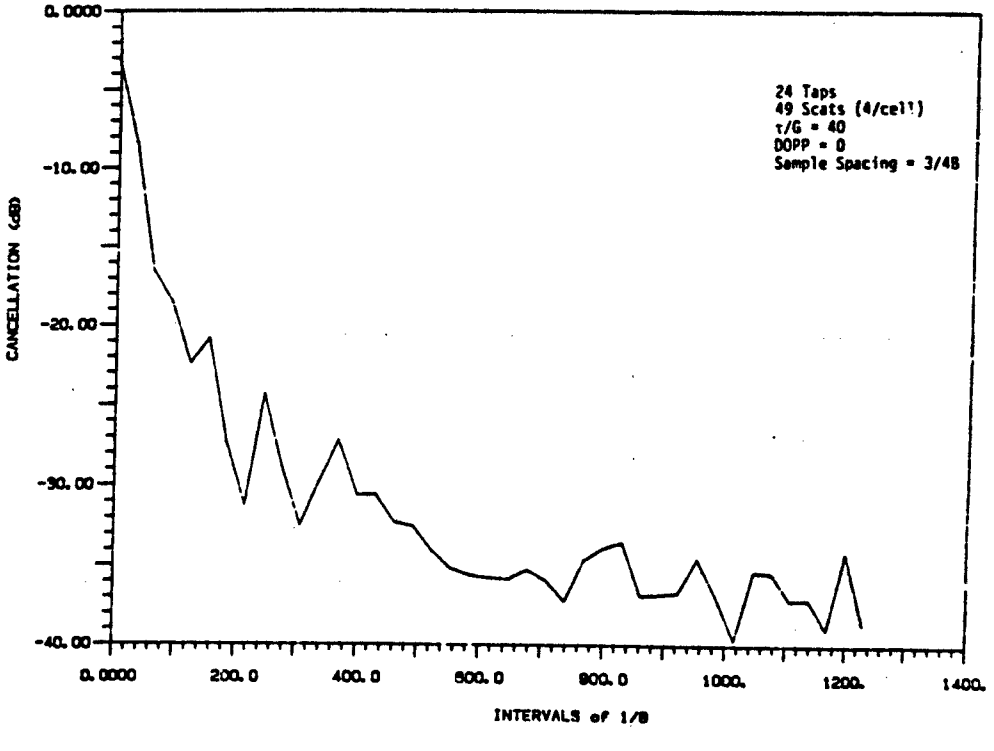

Fig. 18 -- Scatter Canceller - Samples Spaced by .75/B

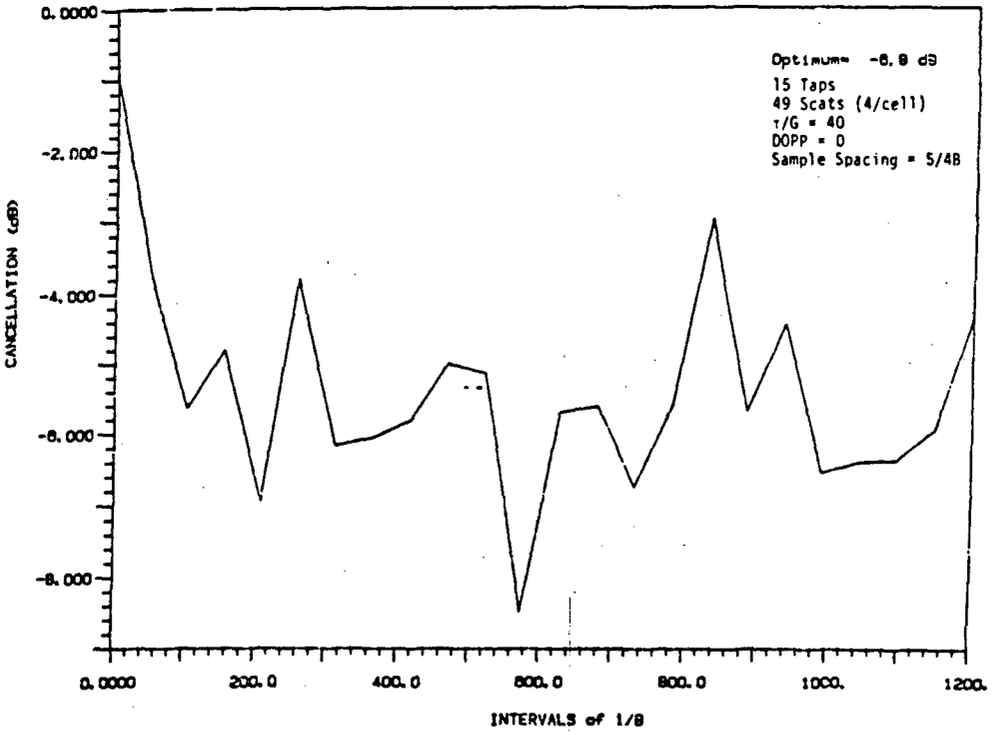

Fig. 19 -- Scatter Canceller - Samples Spaced by 1.25/B

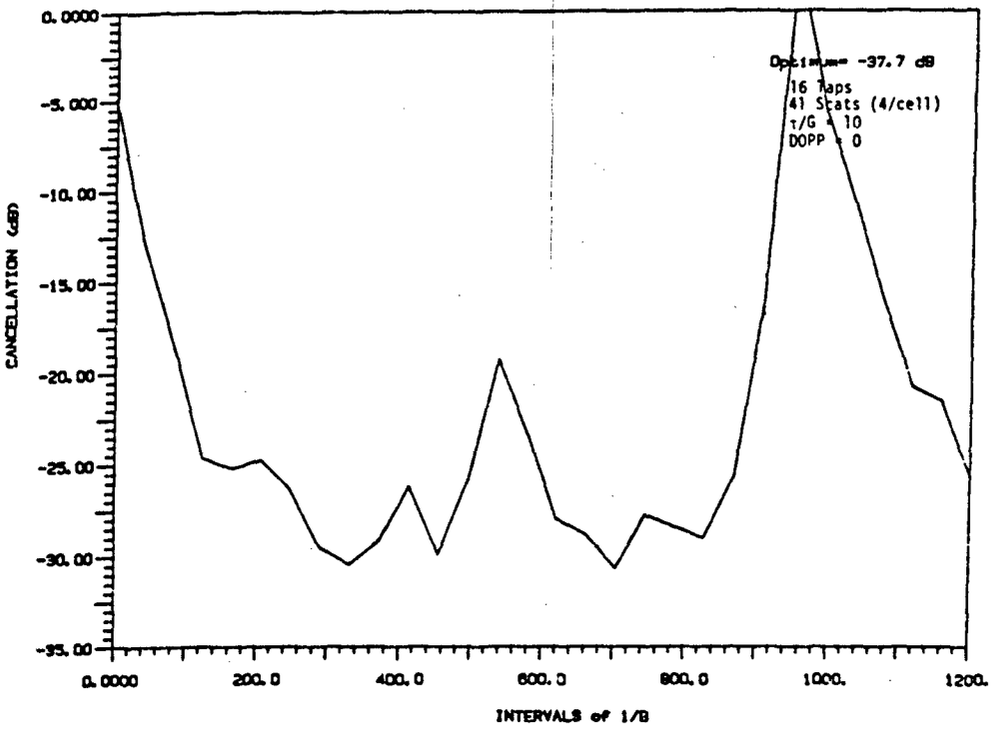

Fig. 20 -- Scatter Canceller with $\tau/G = 10$

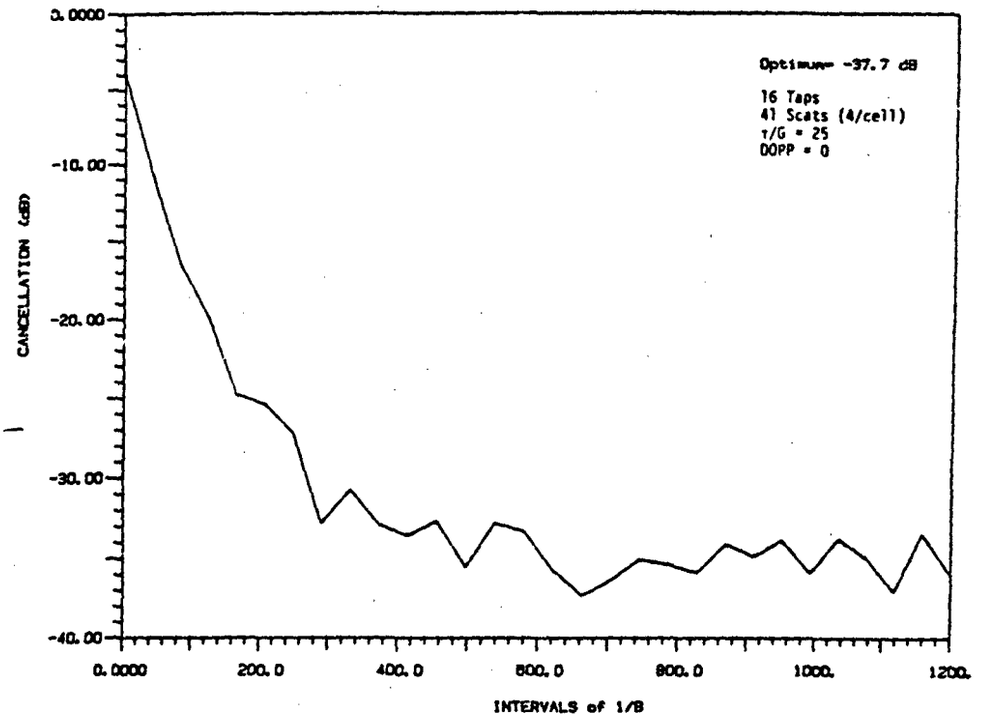

Fig. 21 -- Scatter Canceller with $\tau/G = 25$

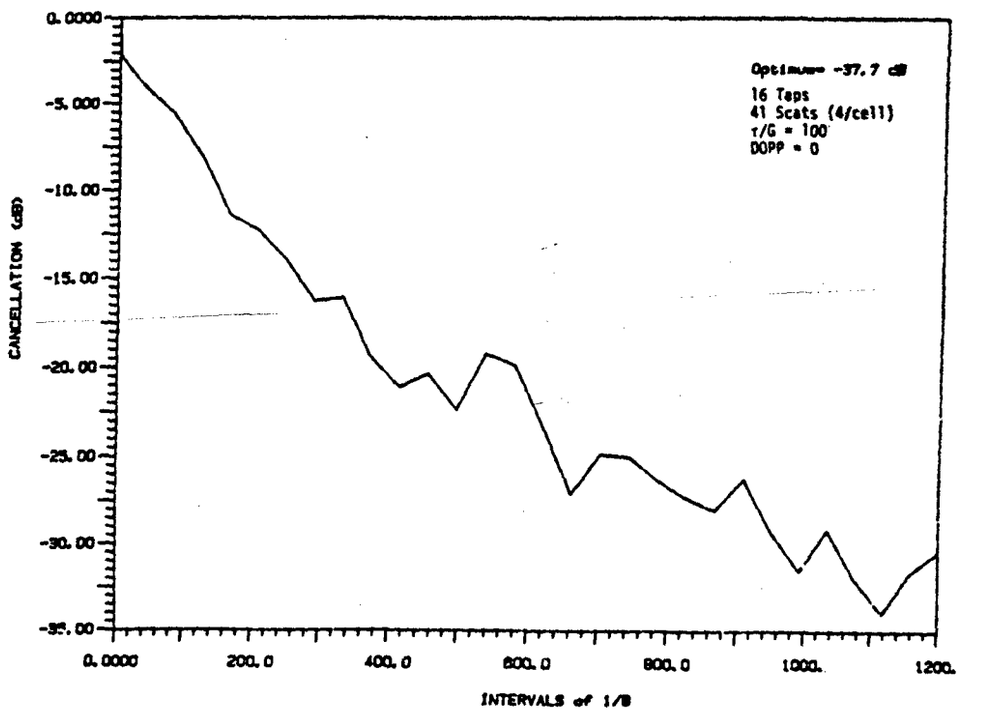

Fig. 22 -- Scatter Canceller with $\tau/G = 100$

a cosine-frequency spectrum of the jamming. Convergence properties of this weight updating algorithm may be different for other interference spectra. It was also shown that 4 scatterers per interval of 1/B suffice to represent distributed scatterers in the simulation and that a sample spacing of 1/B is small enough to provide good scatter cancellation. A τ/G ratio of 25 appears to be a good choice of parameters in Fig. 4 for weight updating, since it yields rapid convergence without excessive weight jitter.

The preceding examples have dealt with the stationary case, i.e., where there is no doppler offset due to motion. The next section contains examples with motion where the optimum weights are continuously changing.

5. EFFECT OF DOPPLER OFFSET DUE TO MOTION

In many situations where jamming is scattered into the receiving beam of a radar or communication system, the jammer, scattering medium, or receiving antenna may be moving. This causes a relative doppler offset between the jamming signal received directly by the auxiliary antenna and the scattered jamming entering the main receiving beam. In Fig. 1, let

$$u = \frac{d}{dt}\left[(R_1+R_2-R_d)/c\right] \tag{7}$$

When the main channel and auxiliary outputs are sampled at intervals of 1/B, the sample to sample phase shift in cycles due to this doppler offset is

$$\Delta\phi = \frac{u}{\lambda B} \tag{8}$$

where λ is the wavelength. This intersample phase shift is input to the simulation program as the parameter DOPP. The velocity of a scattering chaff cloud will generally be low compared to the velocity of an airborne jammer or airborne receiving antenna. A typical value for u due to aircraft motion is between 100 and 300 meters/sec. Typical values of wavelength range from .03 meters to 1 meter and typical bandwidth from 1 to 5 MHz. Table 1 shows the parameter DOPP for various combinations of platform velocity, wavelength, and bandwidth. Note that DOPP is less

than .001 in most of the examples. In the worst case of maximum velocity, minimum wavelength and low bandwidth, DOPP approaches .01.

Table 1 Typical Values of Intersample Phase Shift

u(meters/sec)	(meters)	B(MHz)	DOPP(cycles)
100	.033	1	.003
100	.033	5	.0006
100	.1	1	.001
100	.1	5	.0002
100	1	1	.0001
100	1	5	.00002
300	.033	1	.009
300	.033	5	.002
300	.1	1	.003
300	.1	5	.0006
300	1	1	.0003
300	1	5	.00006

The examples of Figs. 23 through 25 show the effect of motion on the cancellation ratio as the parameter DOPP is increased. The parameters in these three samples are the same as in Fig. 21, viz., 16 taps, 10 intervals of 1/B contain scatterers, and $\tau/G = 25$. With no motion (Fig. 21), the optimum cancellation ratio is 37.7 dB and the adaptive system achieves 35 dB of cancellation after about 400 iterations. When the motion parameter DOPP is increased from 0 to .0003 in Fig. 23, the performance degradation due to motion is very small. Approximately 32 dB

of cancellation is obtained and the convergence rate is the same. In Fig. 24, DOPP is .001 and the cancellation ratio is reduced to about 23 dB. With DOPP=.002 in Fig. 25, approximately 18 dB of scatter cancellation is achieved.

The effect of motion on scatter cancellation for a larger system with 35 delay taps is illustrated in Figs. 26 through 31. In each case, 25 delay intervals of width 1/B contain scatterers, with 4 Rayleigh scatterers per cell. Both the jamming samples and scatterer coefficients are identical in the 6 examples, except for the doppler shift of the jamming signals in the last 5 cases. The time constant used in weight updating, τ/G, is 25 in each case. With no motion in Fig. 26, the optimum cancellation ratio based on the sample covariance matrix is 38 dB. The cancellation ratio in the simulation reaches 30 dB after roughly 400 weight iterations. A relative motion corresponding to a DOPP of 10^{-4} cycles per interval of 1/B, Fig. 27, yields a transient response almost identical to that with no motion. In Fig. 28, with DOPP=.0003, the adaptive system again achieves about 30 dB of cancellation. When DOPP is increased to .001 in Fig. 29, the cancellation ratio is reduced to about 23 dB. With a doppler offset of .002 samples per cycle, Fig. 30, the cancellation ratio is ~17 dB. In Fig. 31, with DOPP=.005, a value larger than most of the typical cases of Table 1, the cancellation ratio is only ~8 dB.

These simulation results show that, for most typical doppler offsets of Table 1, the simple weight updating algorithm for a scatter canceller converges rapidly enough to follow the doppler rotation of weights. In cases of higher doppler offset, there are possible modifications of the

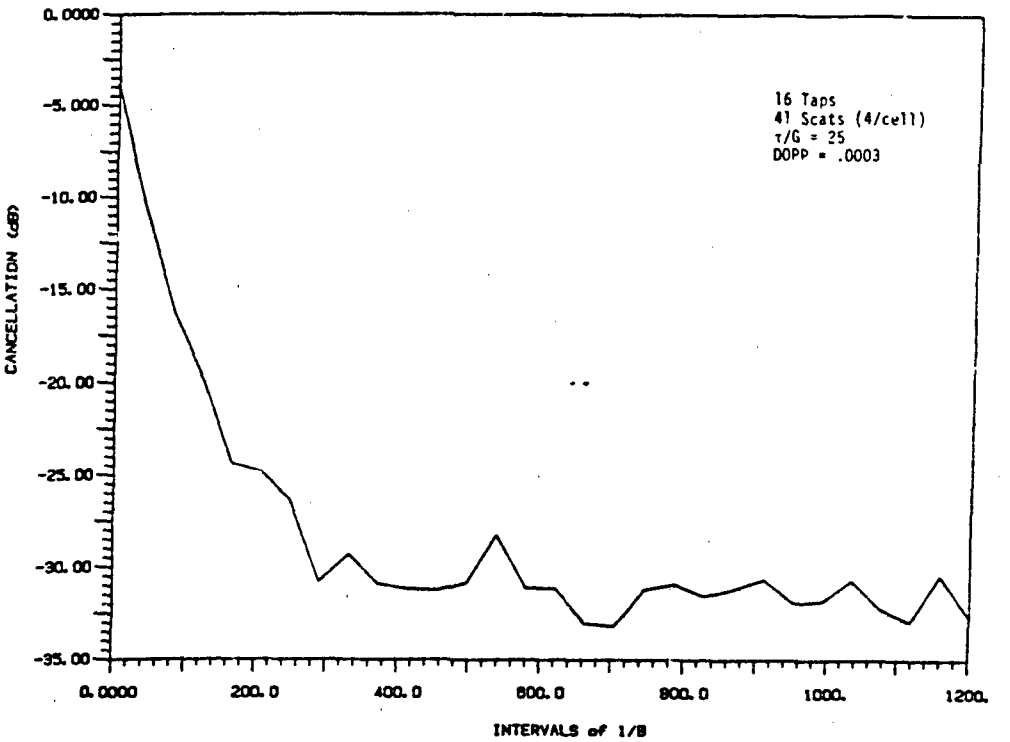

Fig. 23 -- Scatter Canceller with Motion - .0003 cycles/sample
(Compare Fig. 21.)

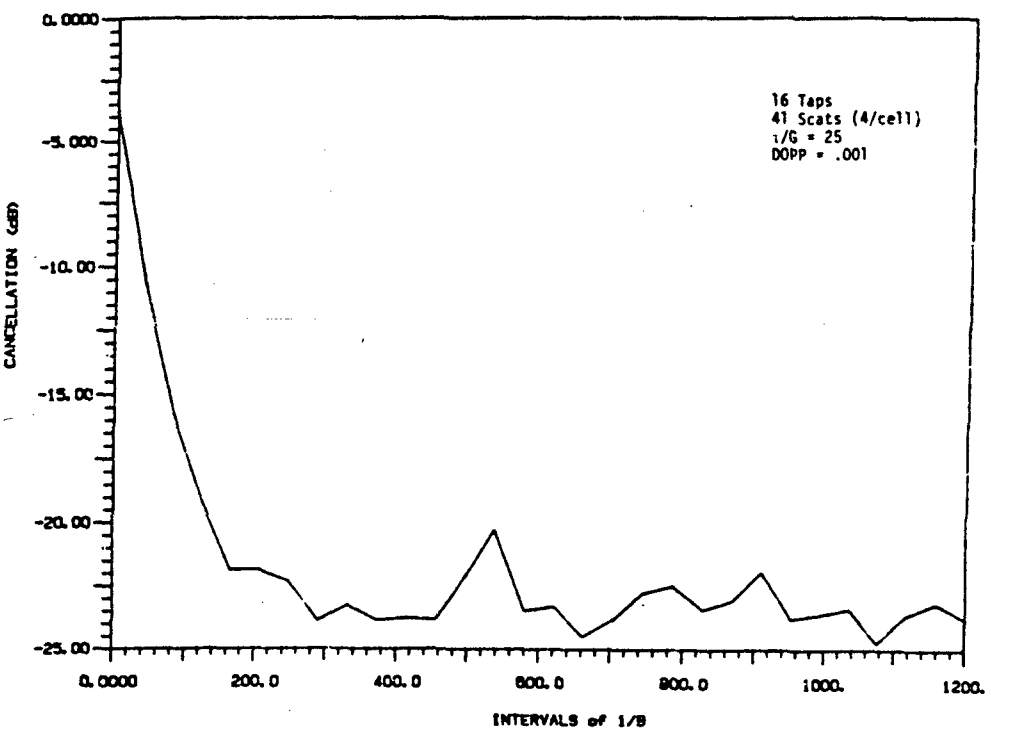

Fig. 24 -- Scatter Canceller with Motion - .001 cycles/sample

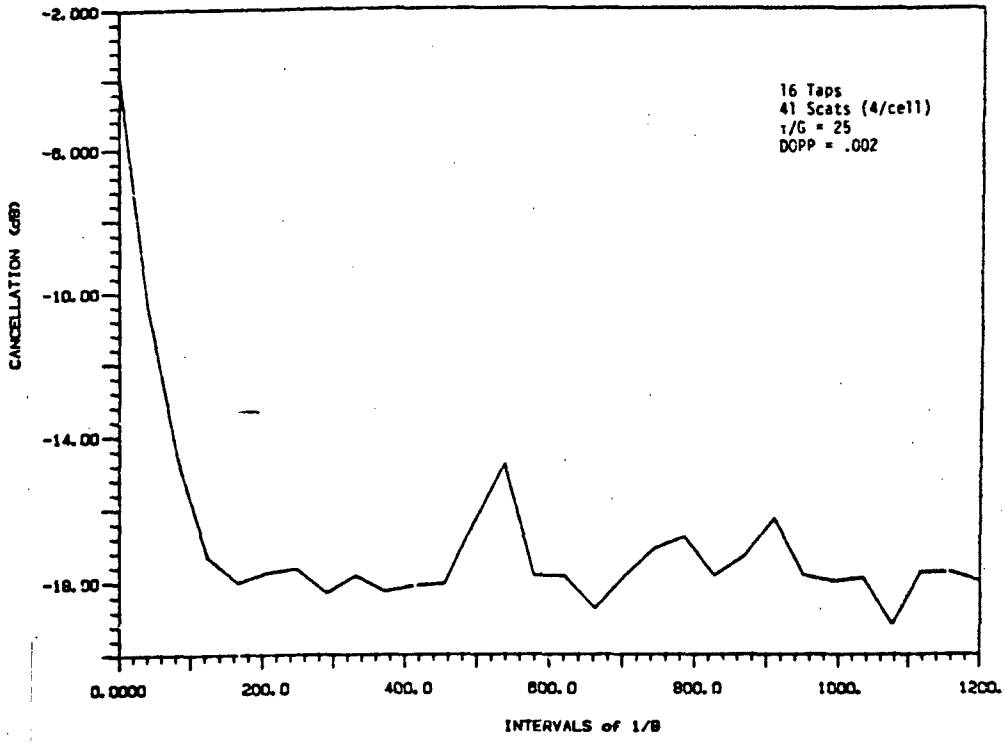

Fig. 25 -- Scatter Canceller with Motion - .002 cycles/sample

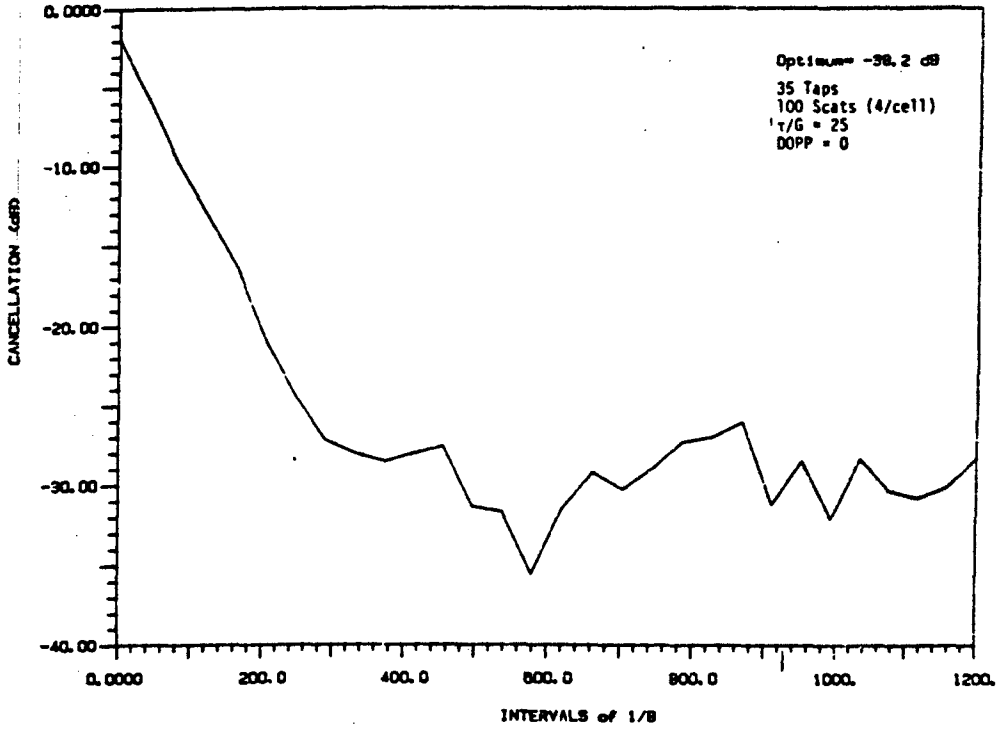

Fig. 26 -- Scatter Canceller - Reference Case of No Motion

scatter canceller to permit doppler tracking. One possibility is to try a set of different additive weight rotation rates to find one that yields good cancellation. Since the weights converge in less than 1 millisecond in most cases, the required search time would be small. Another possiblity, which was discussed in an earlier report on this contract, is the use of gradient technique which controls both the weights and the weight rotation rate. When two or more moving interference sources are illuminating the scattering region, however, a single weight rotation rate will not provide the necessary compensation for motion. In these cases, rapid convergence of the weights provide the best solution. It is encouraging that, for most typical cases of interest, the simulation results show that 20 dB or more cancellation can be achieved with the simple weight updating algorithm of Eq. (4).

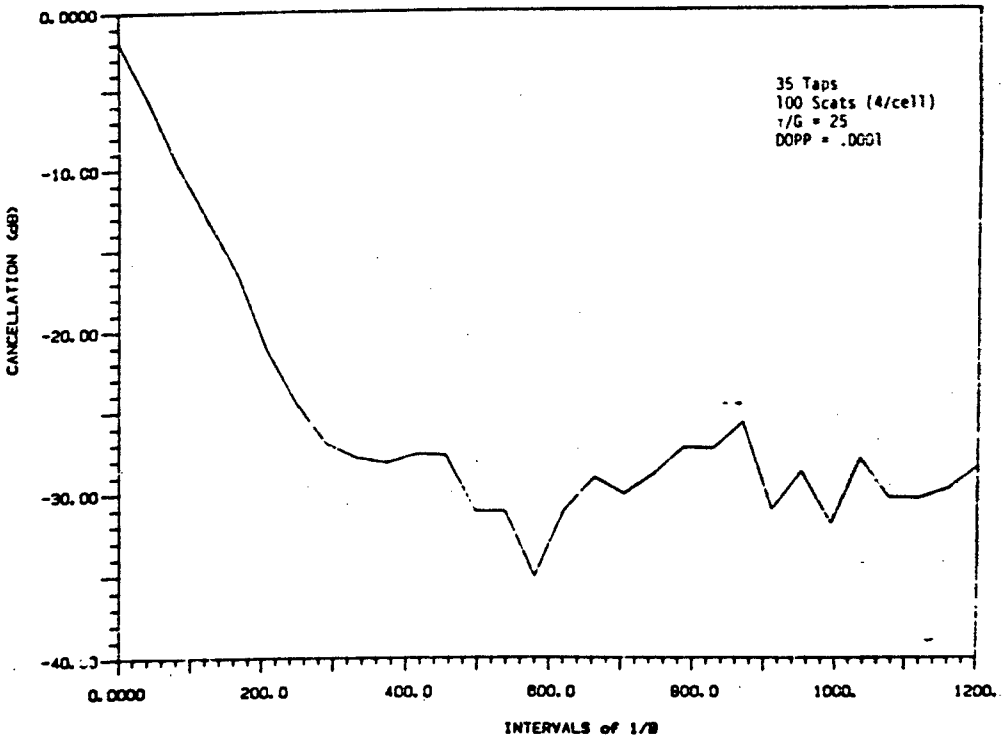

Fig. 27 -- Scatter Canceller - DOPP = .0001

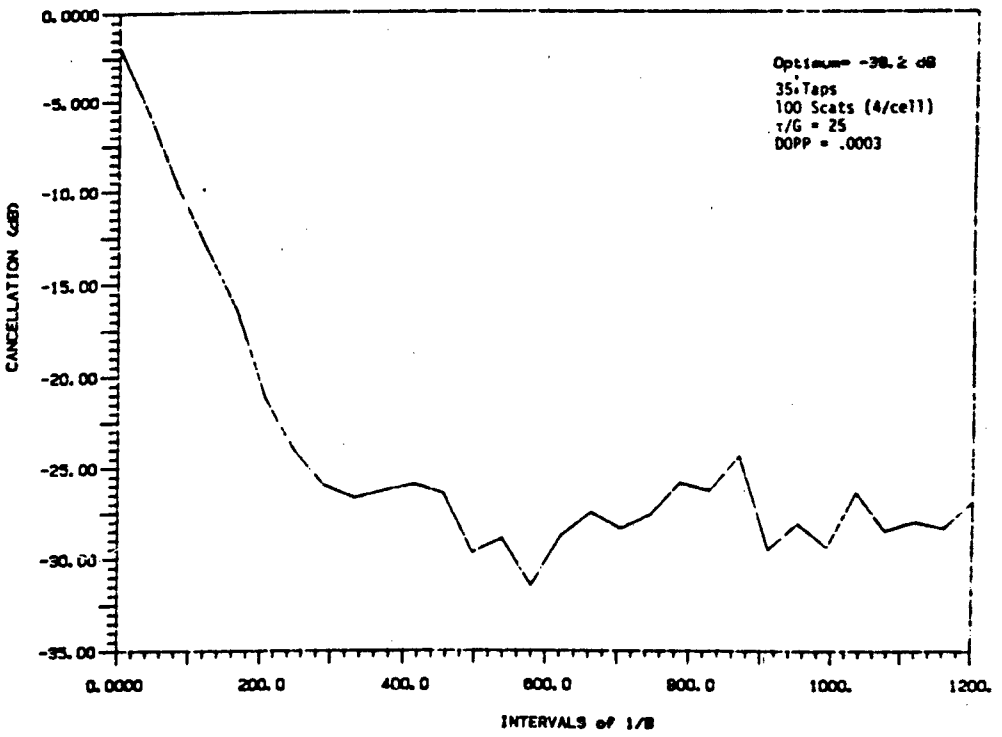

Fig. 28 -- Scatter Canceller - DOPP = .0003

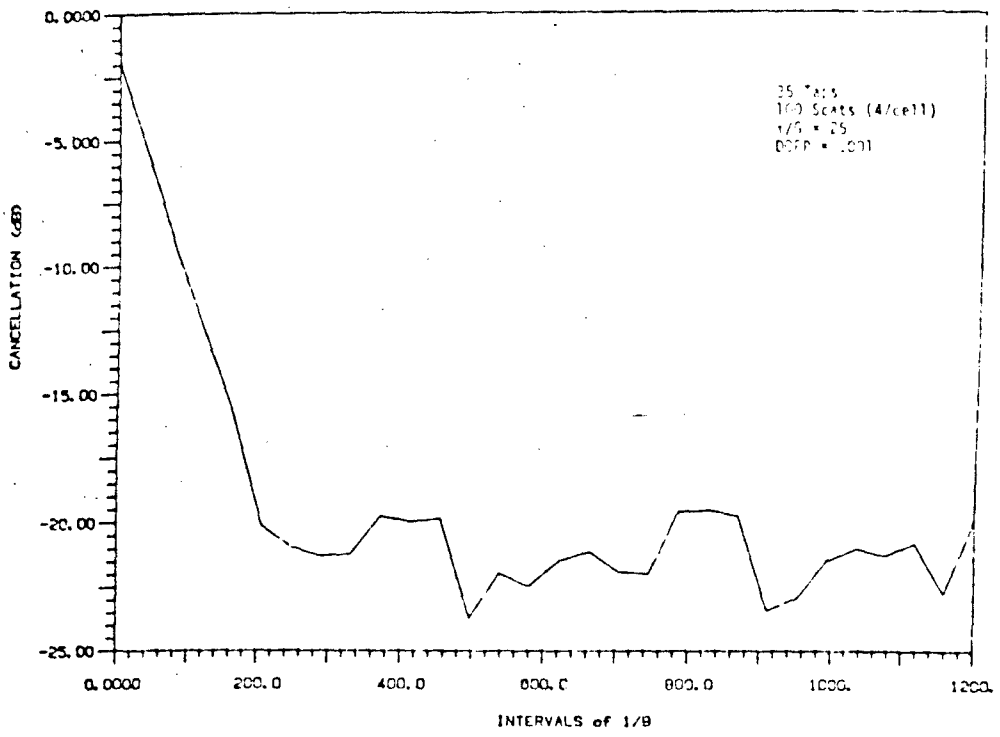

Fig. 29 -- Scatter Canceller - DOPP = .001

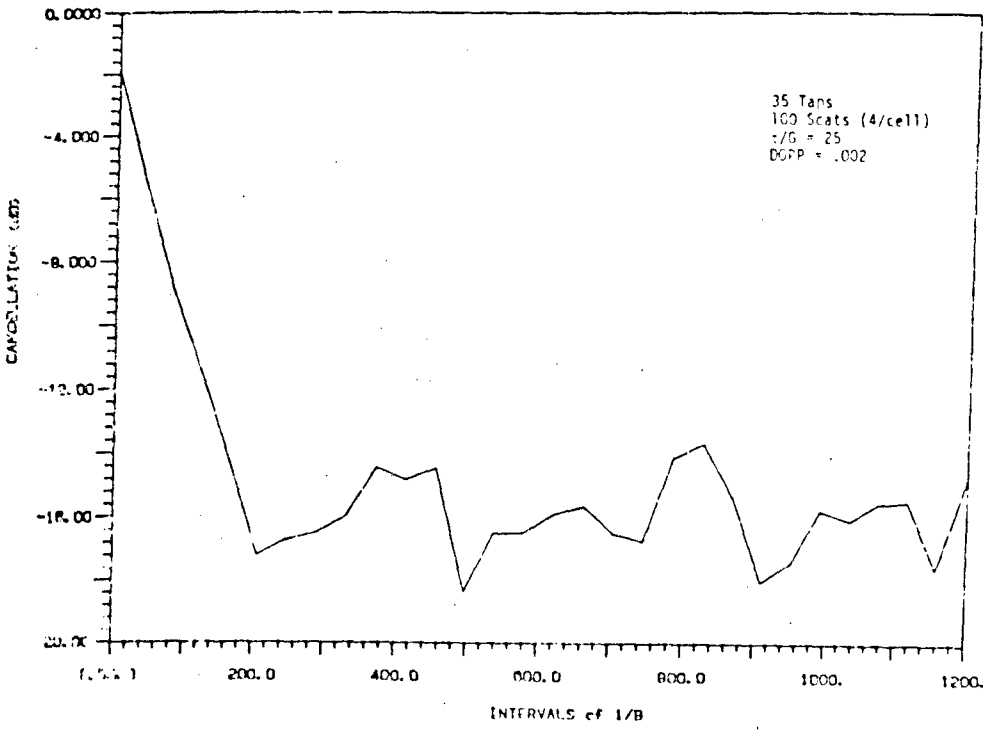

Fig. 30 -- Scatter Canceller - DOPP = .002

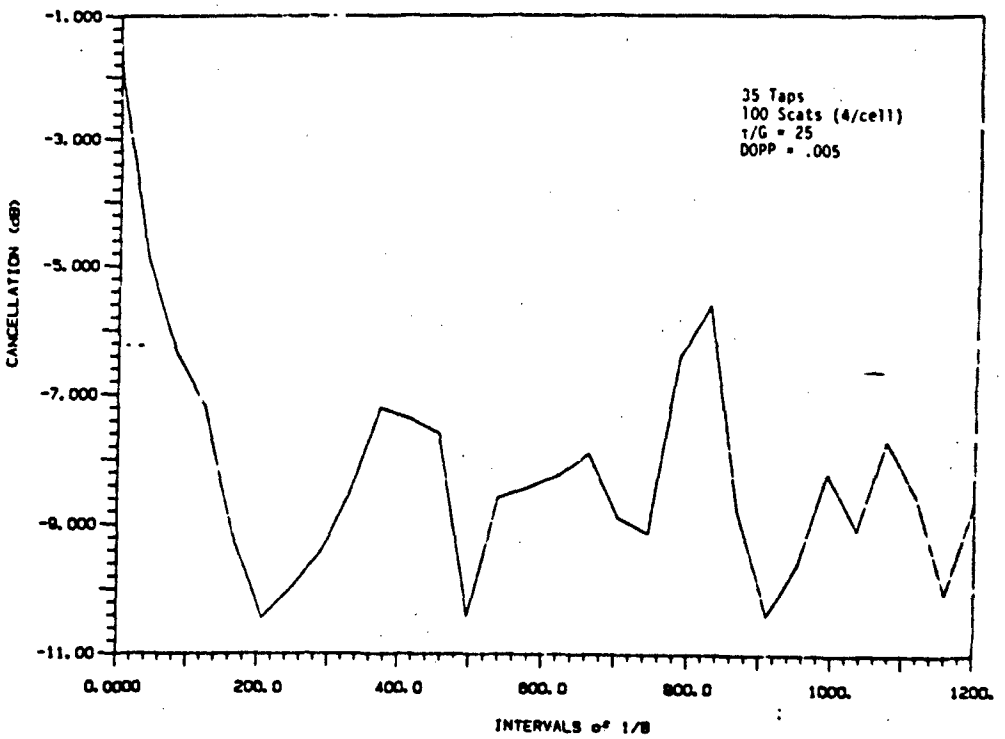

Fig. 31 -- Scatter Canceller - DOPP = .005

6. CONCLUSIONS

Scattered jamming which enters the main beam of a radar or communication system can limit the performance of future systems with low sidelobe antennas. A method of adaptively cancelling scattered interference has been described and simulated in detail. When the relative delay between the scattering path and the direct LOS path to the receiving antenna covers a large interval relative to the range resolution (or reciprocal bandwidth) of the system, a large number of adaptive weights are required in the scatter canceller. A simple algorithm for efficiently computing the scatter canceller weights has been defined and simulated. It was shown that the adaptive weights generated using this algorithm converge quickly to values which provide a high level of scatter cancellation. Using this method of weight computation, the adaptive system can follow the doppler offset due to motion of an airborne jammer or airborne receiving antenna in most cases of interest.

APPENDIX A

SIMULATION PROGRAMS

APPENDIX A

SIMULATION PROGRAMS

1. INTRODUCTION

The over-all simulation naturally divides into a few functional blocks. Mainly because of the limited program storage (somewhat under 28K words) of the DEC 11/23, it seemed wise to realize these functions as separate programs. A second reason for this organization is that the early segments of the simulation tend to be used very little compared to the final segment; the early segments produce intermediate date which are used over and over with different parameters and in different combinations.

The organization of the simulation is depicted in Fig. A-1. Squares represent functional program blocks, labeled with their names. The links between blocks are labeled with the data files that communicate between them. A few input parameters are also usually required for any segment; these are described where the separate blocks are discussed. Generally, all the pertinent parameters are reproduced and accumulated in successive output files.

2. FILTER COEFFICIENTS FOR JAMMING GENERATOR

The zero-frequency, symmetrical bandpass jamming used in the simulation is generated by filtering Gaussian white noise generated by subroutine RAN3. The coefficients constituting the symmetrical impulse response of the filter are computed by the programs CS.FTN or KO.FTN and output to a file x.WTS. The program CS uses a cosine-squared spectrum; KO a cosine spectrum. Either of these programs requires three inputs. The first input is a 4-character label, denoted x, to identify the case and the two output files

written. The other two are the parameters IS and DBTAIL. IS is the number of samples per one-over-bandwidth time interval, 1/B, at which the jamming will be generated. More than one sample per 1/B is used to enable simulation of more than one random scatterer per 1/B interval. DBTAIL is the parameter which determines what fraction of the jamming power is ignored by restricting the filter impulse response to a finite (symmetrical) interval. Usually DBTAIL=-40 dB has been used. CS or KO determines how many coefficients are required to insure the condition is met. The file x.CSQ is output by CS and x.COS by KO. These files contain various running terms of the approximation and can be printed out to check on the program.

The main output of CS or KO is the file x.WTS containing the impulse response coefficients for the filter. This file also contains, at the end, the theoretical autocorrelation for the filter being realized, evaluated at intervals of 1/(IS*B). These programs will ordinarily be used only once at the start of a series of simulation runs.

3. JAMMING GENERATION

The program JAM generates a long file of jamming. Its inputs are the file x.WTS and the parameters NSAM and NRDN. NRDN is the number of times to call RAN3 before starting; this allows the possibility of producing different random jamming runs with the same parameters. NSAM is the number of samples of jamming desired. The jamming is generated and written in blocks of 50 complex random samples, so the number of samples actually produced is the nearest multiple of 50 greater than or equal to NSAM. This jamming output is written to a binary file x.JAM. If a very long file is written, its early segments can be used for shorter runs, but no provision

has been made for skipping any part of the jamming (easily added, however). The first record of the file x.JAM contains the accumulated parameters so far specified, as well as the filter coefficients used.

4. CKJAM

This program was used during early development to verify that the coefficients used for the filter actually produced a random process having the desired correlation function. It reads the first (parameter) record of the file x.JAM and writes the file x.COR containing the autocorrelation function actually produced by the truncated impulse response used for the filter. This may be compared to the theoretical autocorrelation function which was output to the files x.CSQ or x.COS by CS or KO for verification.

5. SCATTERER GENERATION

The two programs RANSC or GRANSC generate a block of random scatterers for use by the simulation. Inputs required are a 4-character name, y, and the parameters MINDEL, MAXDEL, INCR. MINDEL is the delay to the first scatterer, MAXDEL is the delay to the last, or just beyond it, and INCR is the delay increment, normally 1, between scatterers. Either program writes an output file containing NSC,(ND(I),CR(I),I=1,NSC) as I4/(2F8.3). NSC is the number of scatterers produced; ND(I) and CR(I) are the delay, in samples, and the complex reflection coefficient for the I-th scatterer.

Between them, the files x.JAM and y.REF provide the random jammer outputs and the random band of scatterers required for the simulation. If RANSC is used the CR(I) have random phase but unit magnitude. If GRANSC is used the CR(I) are random complex Gaussian with unit expected magnitude squared.

6. THE SIMULATION PROPER

Several input parameters are required by the simulation and figure in the input list at the start. They are:

RUN(1): the 4-character name, z, to use as a label on the output file, z.SIM, and on a plot of cancellation ratio versus time.

FJAM(1): the 4-character identification, x, of the jamming file, x.JAM, to use.

FREF(1): the 4-character identification of the file y.REF of reflectors to use.

NT: the number of delay line taps to use in the canceller, hence the the number of canceller feedback loops to mechanize.

NBSO: the number of extra delays of amount 1/B to add to all the scatterer delays, ND(I), in the file y.REF when they are used. The taps always start at delay=0, so this permits the tap delays to precede in time the first scattered jamming samples, if desired.

KQ: the sample spacing between delay line taps. Ordinarily it will be the same as IS in Section 2.

LAPP: the number of sample steps to advance the whole process between updates of the canceller loops. It too will ordinarily be the same as IS.

IAVE: the number of output residues over which to average for the purpose of estimating the residue power. This is the quantity plotted versus time. If IAVE=1 the raw power of each sample is plotted; however, at present there is not enough variable storage to hold a very long run if IAVE=1.

INBLEN: the length of time to run the simulation,measured in units of 1/B.

INBDOP: the length of time to run the simulation, in units of 1/B, before turning on the Doppler shift, if there is any.

POW: a previously obtained estimate (possibly) of the jamming power in the file x.JAM. If the value entered as input is less than or equal to zero then before the simulation starts, the simulation program will read through the file x.JAM and estimate POW. If a positive POW is given as input, this estimation phase will be skipped and that POW used. The value of POW is used in any run to scale the total scatterer jamming power to unity (by modifying the reflection coefficients). This makes the theoretical uncancelled scattered jamming level 0 dB. TAU and GAIN are the canceller loop time constants and gains.

DOPP: the Doppler shift to superimpose on the scattered jamming, measured in cycles per 1/B, hence per IS samples.

After the simulation program has read in the first or parameter record of the x.JAM file, it proceeds to fill a working buffer, H(1000), with the complex jamming. A pointer (KS in the program) moves along this buffer in steps of LAPP samples. The total scatter and the tap outputs move along with this pointer. Fig. A-2 should help one to visualize the process. When the pointer, KS, passes 1000 the upper 500 samples are shifted back to the lower 500 H cells, a new set of 500 samples is read in to the upper 500 cells, or as much data as are available in the x.JAM file, the pointer is decremented 500, and the simulation continues.

At the end of the run the plotting routines are called. If some of the arrays are to be enlarged, for example to allow long runs with IAVE=1, this would be a good place to divide the program again and leave a separate plotting program as the last step.

There are some other program outputs, including theoretic optimum performance based on the sample covariances, which should be more or less self-explanatory. The programs are extensively commented, so that most questions which might arise should be answerable by examining the programs themselves.

An attempt has been made to shortstop inconvenient parameter choices, chiefly those that lead to array overflows, but absolute guarantees are lacking.

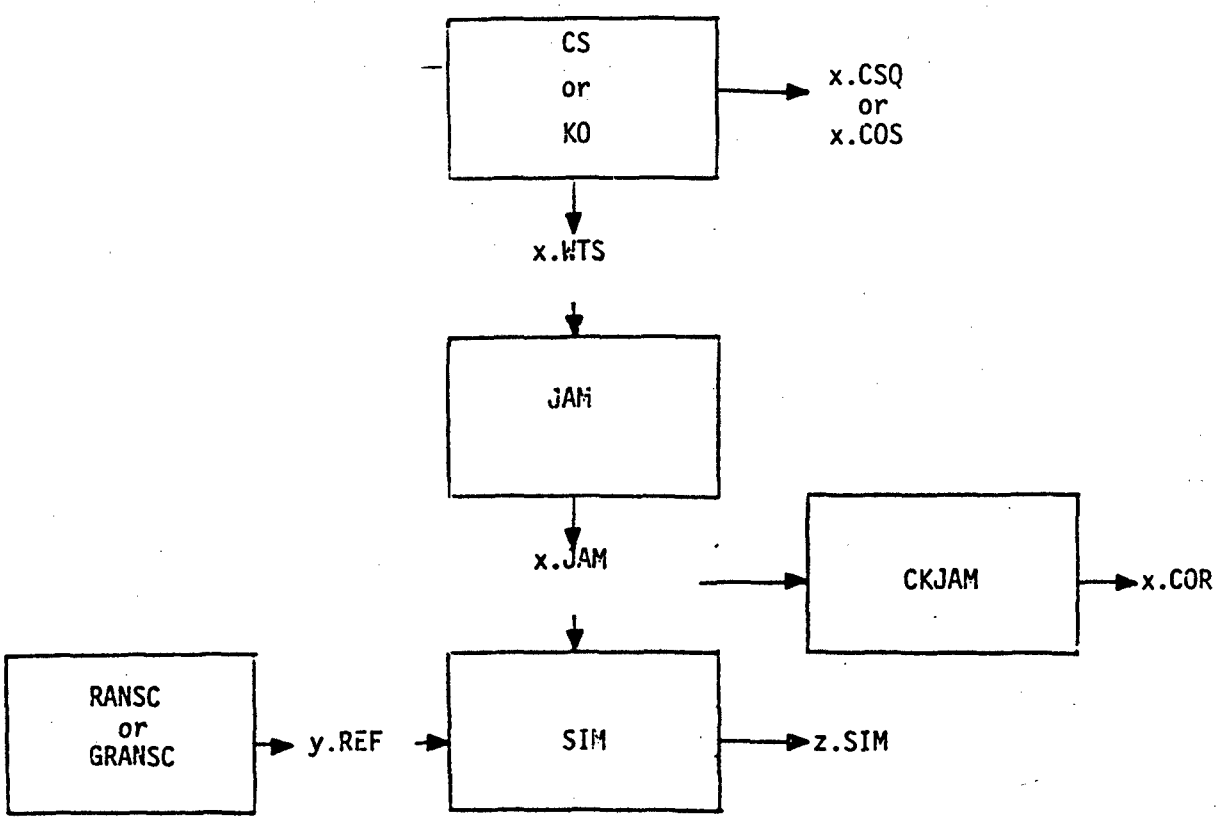

Fig. A-1 Organization of Simulation Programs and Files

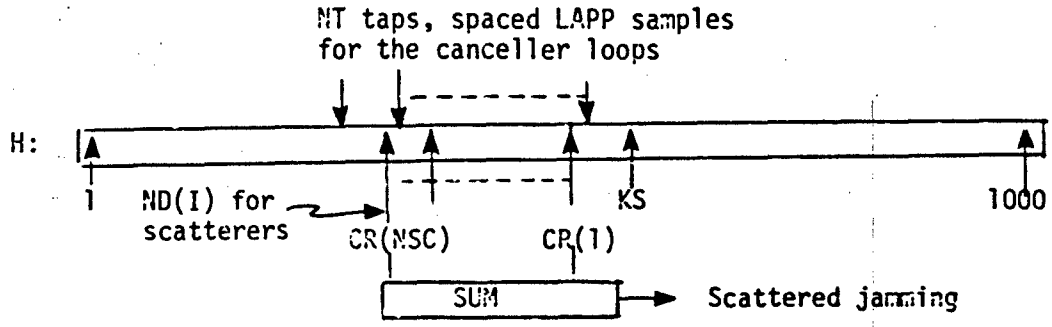

Fig. A-2 Arrangement for Processing Data

```
      PROGRAM COSQWT
CCC   Calculates an impulse response for Cosine-squared spectrum
CCC   and writes a file suitable for input to a noise generator program.
CCC   See statement at 44 FORMAT for file contents.
CCC   Weights truncated where average tail power is at least DBTAIL
CCC   dB down from average power of part used.
      REAL TER(303),SUM(303),EMB(303),HEAD(8),FILE(2),QUE(303),PRIN(2)
      REAL COR(607)
      DATA PI,HEAD,PRIN(2)/3.1415926,'COSQ',' WTS',6*'    ','.CSQ'/
      DATA FILE(2)/'.WTS'/,QUARPI/.7853982/
      CALL DATE(HEAD(4))
      CALL TIME(HEAD(7))
      S=1.
      TYPE 40
40    FORMAT(' Enter FILE-NAME, I-PARAMETER, DB-TAIL as 1X,A4,I3,F6.0')
41    FORMAT(1XA4,I3,F6.0)
      ACCEPT 41,  FILE(1),IS,DBTAIL
      IF(DBTAIL.GT.0.)DBTAIL=-DBTAIL          ! IF FORGET MINUS
      TYPE 41,  FILE(1),IS,DBTAIL             ! VERIFY ON 'SCOPE.
      PRIN(1)=FILE(1)
      CALL ASSIGN(2,PRIN,8,'NEW')
      WRITE(2,42)1,29,FILE(1),HEAD(3),HEAD,2,30       ! FOR IP-125 ONLY
42    FORMAT(/14A4)
      WRITE(2,45)IS,DBTAIL
45    FORMAT(15X'M'7X'TERM'10X'SUM'9X'EMB'4X'I='I3,5X'DBTAIL='F6.0/)
      FERR=10.**(DBTAIL/10.)
      EYE=IS
      U=2.*((.5*EYE)**4)/(3.*FERR)
      CALL ASSIGN(1,FILE,8,'NEW')     ! MAY ND TO CHGE FOR 11/23.
      DO 1 I=1,303
      EM=I
      V=1.-(2.*EM/EYE)**2
      IF(V)2,3,2
3     Q=QUARPI
      GO TO 5
2     Q=COS(PI*EM/EYE)/V
5     QUE(I)=Q
      TER(I)=Q**2
      S=S+TER(I)*2.
      SUM(I)=S
      EMB(I)=(U/S)**(1./3.)
1     IF(EM.GE.EMB(I))GO TO 7
7     M=I
      IF(M.GE.303)STOP          'COSQWT:  DIMENSIONS 303 TOO SMALL, ALSO 607
      QUENUL=1./SQRT(S)         ! Renorm so E[|y|**2]=1.
      DO 8 I=1,M
8     QUE(I)=QUE(I)*QUENUL
      WRITE(2,43)(I,TER(I),SUM(I),EMB(I),I=1,M)
```

```
43      FORMAT(I16,3F13.6)
CCC     FILE(1) is case identifier.  IS is parameter i.
CCC     DBTAIL is rel mag of cut off tail, average power.
CCC     QUENUL is weight K-sub-zero.  QUE(I) are rest of weights.
        WRITE(1,44)FILE(1),IS,DBTAIL,M,QUENUL,(QUE(I),I=1,M)
44      FORMAT(1XA4,I3,F6.0,I4,(F12.6))
        J=M+M+1          ! Calculate correlation function at 1/(IS*B) steps.
        DO 4 I=1,J
        EM=I
        EM=EM/EYE
        IF(I-IS)6,4,6
6       COR(I)=SIN((PI*EM))/(PI*EM)/(1.-(EM)**2)
4       CONTINUE
        COR(IS)=.5
        J=MIN0(J,64)             ! Cut printout to 64 values for now.
        WRITE (1,46)(COR(I),I=1,J)
46      FORMAT(/(8F10.6))
        CALL CLOSE(1)
        STOP    'COSQWT'
        END
```

```
COMPLEX FUNCTION RAN3(I1,I2)
R1=RAN(I1,I2)
R2=RAN(I1,I2)
A=6.2831853*R2
RAN3=SQRT(-ALOG(R1))*CMPLX(COS(A),SIN(A))
RETURN
END
```

```
      PROGRAM KOWT
CCC   Calculates an impulse response for Cosine spectrum
CCC   and writes a file suitable for input to a noise generator program.
CCC   See statement at 44 FORMAT for file contents.
CCC   Weights truncated where average tail power is at least DBTAIL
CCC   dB down from average power of part used.
      REAL TER(303),SUM(303),EMB(303),HEAD(8),FILE(2),QUE(303),PRIN(2)
      REAL COR(607)
      DATA PI,HEAD,PRIN(2)/3.1415926,'COS ',' WTS',6*'    ','.COS'/
      DATA FILE(2)/'.WTS'/,QUARPI/.7853982/,Q0/3.8239104/
      CALL DATE(HEAD(4))
      CALL TIME(HEAD(7))
      TYPE 40
40    FORMAT(' Enter FILE-NAME, I-PARAMETER, DB-TAIL as 1X,A4,I3,F6.0')
41    FORMAT(1XA4,I3,F6.0)
      ACCEPT 41,  FILE(1),IS,DBTAIL
      IF(DBTAIL.GT.0.)DBTAIL=-DBTAIL          ! IF FORGET MINUS
      TYPE 41,  FILE(1),IS,DBTAIL             ! VERIFY ON 'SCOPE.
      PRIN(1)=FILE(1)
      CALL ASSIGN(2,PRIN,8,'NEW')
      WRITE(2,42)1,29,FILE(1),HEAD(3),HEAD,2,30       ! FOR IP-125 ONLY
42    FORMAT(/14A4)
      WRITE(2,45)IS,DBTAIL
45    FORMAT(15X'M'7X'TERM'10X'SUM'9X'EMB'4X'I='I3,5X'DBTAIL='F6.0/)
      FERR=10.**(DBTAIL/10.)
      EYE=IS
      U=(EYE**3)/FERR
      S=Q0**2
      CALL ASSIGN(1,FILE,8,'NEW')     ! MAY ND TO CHGE FOR 11/23.
      DO 1 I=1,303
      EM=I
      V=EM/EYE-.25
      IF(V)2,3,2
3     Q=PI
      GO TO 5
2     Q=GAMMA(V)*SIN(PI*V)
5     QUE(I)=Q/GAMMA(V+1.5)
      TER(I)=QUE(I)**2
      S=S+2.*TER(I)
      SUM(I)=S
      EMB(I)=SQRT(U/S)
1     IF(EM.GE.EMB(I))GO TO 7
7     M=I
      IF(M.GE.303)STOP          'COSQWT:  DIMENSIONS 303 TOO SMALL, ALSO 607
      S=1./SQRT(S)
      QUENUL=Q0*S      ! Renorm so E[|y|**2]=1 for symm impulse seq.
      DO 8 I=1,M
8     QUE(I)=QUE(I)*S
      WRITE(2,43)(I,TER(I),SUM(I),EMB(I),I=1,M)
43    FORMAT(I16,3F13.6)
```

```
CCC     FILE(1) is case identifier.  IS is parameter i.
CCC     DBTAIL is rel mag of cut off tail, average power.
CCC     QUENUL is weight K-sub-zero.  QUE(I) are rest of weights.
        WRITE(1,44)FILE(1),IS,DBTAIL,M,QUENUL,(QUE(I),I=1,M)
44      FORMAT(1XA4,I3,F6.0,I4,(F12.6))
        J=M+M+1          ! Calculate correlation function at 1/(IS*B) steps.
        J=MINO(J,32)     ! Just enough to check.
        IF(IS.GT.1)COR(IS/2)=QUARPI
        DO 4 I=1,J
        EM=I
        EM=EM/EYE
        IF(2*I-IS)6,4,6
6       COR(I)=COS((PI*EM))/(1.-(2.*EM)**2)
4       CONTINUE
        WRITE (1,46)(COR(I),I=1,J)
46      FORMAT(/(8F10.6))
        CALL CLOSE(1)
        STOP    'KOWT'
        END
```

```
      PROGRAM JAMOUT
CCC   Inputs a weight file and outputs filtered white noise.
CCC   For now, weights are real, corr to a spectrum symm about zero freq.
CCC   Generates nearest 50*N.GE.NSAM samples.
      REAL HEAD(8),CASE(2),Q(303),GUNK(2)
      COMPLEX RAN3,WN(303),CC,BN(50)
      DATA HEAD,CASE(2),GUNK(2)/'MAKE',' JAM',6*'    ','.WTS','.JAM'/
      CALL DATE(HEAD(4))
      CALL TIME(HEAD(7))
      TYPE 40
40    FORMAT(' Enter wt file name, number of samples, NRDN as 1XA4,2I6')
      ACCEPT 41,CASE(1),NSAM,NRDN      ! NRDN is number of pre-calls of RAN3.
      TYPE 41,CASE(1),NSAM,NRDN        ! Echo on 'scope.
41    FORMAT(1XA4,2I6)
      CALL ASSIGN(1,CASE,8,'OLD')      ! MAY NO CHGE FOR 11/23.
      READ(1,43)GUNK(1),IS,DBTAIL,M,(Q(M+2-I),I=1,M+1)          ! Reverse orde
43    FORMAT(1XA4,I3,F6.0,I4,(F12.6))
      IF(M.GT.151)STOP          'JAMOUT: MORE WEIGHTS THAN ROOM; EXPAND Q, WN.
      MP=2*M+1
      DO 9 I=M+2,MP             ! Produce symmetric impulse response.
9     Q(I)=Q(MP+1-I)
      CALL CLOSE(1)
      CALL ASSIGN(1,GUNK,8,'NEW')      ! MAY NO CHGE FOR 11/23.
D     WRITE(3,42)CASE(1),IS,DBTAIL,NRDN,NSAM,MP,(Q(I),I=1,MP)
D42   FORMAT(1XA4,I3,F6.0,3I6,(F12.6))
      WRITE(1)CASE(1),IS,DBTAIL,NRDN,NSAM,MP,(Q(I),I=1,MP)
      I1=0
      I2=0
      DO 1 I=1,NRDN             ! Initialize RAN3.
1     CC=RAN3(I1,I2)
      I11=I1
      I22=I2                    ! Save RAN3 start state.
      DO 2 I=1,MP               !Initial filling with white noise
2     WN(I)=RAN3(I1,I2)
D     QOW=0.
D     WOW=0.
D     DO 22 I=1,1000
D     IF(I.LE.MP)QOW=QOW+Q(I)**2
D22   WOW=WOW+CABS(RAN3(I1,I2))**2
      JAM=0                     ! Counts output.
      NB=1                      ! For BNs
      NW=1                      ! For WNs
D     POW=0.
3     CC=(0.,0.)
      DO 4 I=NW,MP              ! Two sums unless NW=1
4     CC=CC+WN(I)*Q(I-NW+1)
      IF(NW.EQ.1)GO TO 6
      KQ=MP+1-NW
      DO 5 I=1,NW-1
5     CC=CC+WN(I)*Q(KQ+I)
6     BN(NB)=CC
D     POW=POW+REAL(CC)**2+AIMAG(CC)**2
      NB=NB+1
      IF(NB.LE.50)GO TO 7       ! See if BN bin full.
```

```
            NB=1
D           WRITE(3,44)BN
D44         FORMAT(6F12.6)
            WRITE(1)BN
            JAM=JAM+50
            TYPE 41,CASE(1),NSAM,JAM          ! CRT progress report
            IF(JAM.GE.NSAM)GO TO 8
7           WN(NW)=RAN3(I1,I2)                ! Replace oldest WN sample.
            NW=NW+1                     ! Update WN pointer.
            IF(NW.GT.MP)NW=1
            GO TO 3
8           CALL CLOSE(1)
D           POW=POW/JAM
D           WOW=WOW/1000
D           TYPE 44,POW,WOW,QOW
            STOP     'JAM'
            END
```

```
      PROGRAM RANREF
CCC   Fill MINDEL,MAXDEL with unit mag, random phase, complex refl coeffs
CCC   at steps of INCR and write file xxx.REF.
      INTEGER ND(101)
      COMPLEX CR(101)
      REAL FI(2)
      DATA FI(2)/'.REF'/I1,I2/0,0/
      TYPE 40
40    FORMAT(' Enter file name, MINDEL, MAXDEL, and INCR as 1XA4,3I4.')
      ACCEPT 41,FI(1),MINDEL,MA,INCR
41    FORMAT(1XA4,3I4)
      TYPE 41,FI(1),MINDEL,MA,INCR              ! Verify on CRT.
      DO 2 I=1,15
2     X=RAN(I1,I2)
      NSC=0
1     NSC=NSC+1
      X=RAN(I1,I2)*6.2831853
      CR(NSC)=CMPLX(COS(X),SIN(X))
      ND(NSC)=(NSC-1)*INCR+MINDEL
      IF(NSC.GT.101)STOP       'More scatterers than space.'
      IF(ND(NSC).LE.MA)GO TO 1
      NSC=NSC-1                 ! Back up one.
      CALL ASSIGN(1,FI,8,'NEW')
      WRITE(1,42)NSC,(ND(I),CR(I),I=1,NSC)     ! Backscatter data.
42    FORMAT(I6/(I4,2F8.3))
      CALL CLOSE(1)
      STOP    'RANREF'
      END
```

```
      PROGRAM GRAREF
CCC   Fill MINDEL,MAXDEL with Gaussian, random complex refl coeffs
CCC   at steps of INCR and write file xxx.REF.
      INTEGER ND(101)
      COMPLEX CR(101),RAN3,Z
      REAL FI(2)
      DATA FI(2)/'.REF'/I1,I2/0,0/
      TYPE 40
40    FORMAT(' Enter file name, MINDEL, MAXDEL, and INCR as 1XA4,3I4.')
      ACCEPT 41,FI(1),MINDEL,MA,INCR
41    FORMAT(1XA4,3I4)
      TYPE 41,FI(1),MINDEL,MA,INCR              ! Verify on CRT.
      DO 2 I=1,432
2     Z=RAN3(I1,I2)             ! Few cycles of RAN first
      NSC=0
1     NSC=NSC+1
      CR(NSC)=RAN3(I1,I2)
      ND(NSC)=(NSC-1)*INCR+MINDEL
      IF(NSC.GT.101)STOP        'More scatterers than space.'
      IF(ND(NSC).LE.MA)GO TO 1
      NSC=NSC-1                 ! Back up one.
      CALL ASSIGN(1,FI,8,'NEW')
      WRITE(1,42)NSC,(ND(I),CR(I),I=1,NSC)     ! Backscatter data.
42    FORMAT(I6/(I4,2F8.3))
      CALL CLOSE(1)
      STOP     'GRANREF'
      END
```

```
      PROGRAM SIMI                !
      COMPLEX H(1000),CR(100),CC,WST(111),SMIN(111),BE,ROT,TURN
CCC   When change SMO dimensions, change lines marked !**!.
CCC   SVROT not used but takes no space.
      COMPLEX SVROT(888),SMO(36,37)    !**! EY* is SMO(,NT+1).
      REAL BOTH(2664),RES(888)         !**!
      REAL FJAM(2),FREF(2),Q(303),HEAD(8),RUN(8),OUT(50),SAM(2)
      EQUIVALENCE (Q,SMO,BOTH,RES),(SVROT,BOTH(889))  !**!
      EQUIVALENCE (RUN(4),FREF),(RUN(7),FJAM)
      DATA HEAD,RUN/'SIMI',7*'    ','----','.SIM',2*'----','.REF',
     .2*'----','.JAM'/RES(1),SAM(1),TWOPI/0.,0.,6.2831853/
      DATA NDIM/36/    !**!
CCC   H is segment of jammer output.  H(1) is oldest; H(1000) newest.
CCC   CR are complex refl coeffs of scatterers. WST conjugate tap weights.
CCC   ND(I) is I-th scatterer delay in samples along the H segment.
CCC   It represents multiples of dt= 1/(IS*B).  Expect all ND(I).GE.0
CCC   NBS0 is nr of 1/B to add to all back scatterer delays.
CCC   All ND(I) increased by IS*NBS0.
CCC   This allows first tap delay (zero) to precede jamming by that much.
CCC   IAVE is nr samples averaged over for cancellation ratio estimate.
CCC   DOPP phase shift is fraction of a cycle per 1/B, hence per IS samples.
CCC   INBDOP is number of 1/Bs to run before turn on Doppler.
CCC   INBLEN is number of 1/Bs to run simulation.
CCC   IT is nr of iterations (IT*LAPP/IS= nr of 1/Bs); IT increments by IAVE.
      INTEGER ND(100)
      LOGICAL*1 MORE
2     TYPE 40
40    FORMAT(/' Enter RUN name, JAM, REF files, NT,NBS0,KQ,LAPP,IAVE,
     .INBLEN,INBDOP'//'   POW,TAU,GAIN,DOPP
     .as 1X3A4,7I6/F10.6,2F10.0,F10.6'
     .//4X'FIRST change paper if plotting directly.  ?? to quit.'/)
      ACCEPT 41,RUN(1),FJAM(1),FREF(1),NT,NBS0,KQ,LAPP,IAVE,INBLEN,
     .INBDOP,POW,TAU,GAIN,DOPP
41    FORMAT(1X3A4,7I6/F10.6,2F10.0,F10.6)
      IF(NT.GT.111)STOP        'More than 111 taps. Chge WST, SMIN dims.'
      DECAY=EXP(-1./TAU)
      DEL=(1.-DECAY)*GAIN
CCC   TAU is A-loop time constant in samples.  GAIN is A-loop gain.
CCC   NT is # of delay line taps.  First tap always zero delay.
CCC   IS (see JAMOUT) is number of samples per 1/B.
CCC   KQ is number of samples per tap (Steps along  H).
CCC   LAPP is number of samples to step between  Applebaum updates.
CCC   Normally IS=KQ=LAPP.
CCC   I-th tap gets the sample H(KS-KQ*(I-1)).
CCC   DOPPTC is Doppler averaging time constant in thousands of 1/Bs.
CCC   POW is average power of beam output.  If POW.LE.0., then
CCC   entire .JAM file will be read and POW calculated.
      CALL ASSIGN(1,FREF,8)
      CALL ASSIGN(3,RUN,8)
      CALL DATE(HEAD(4))
      CALL TIME(HEAD(7))
      WRITE(3,43)HEAD,RUN
43    FORMAT(/1X8A4,6X8A4)
      READ (1,42)NSC,(ND(I),CR(I),I=1,NSC)! Get back-scatterer data.
```

```
42      FORMAT(I6/(I4,2F8.3))
        IF(NSC.GT.100)STOP  'SIMI: MORE SCAT THAN ROOM.  EXPAND ND, CR.'
        CALL CLOSE(1)
        CALL ASSIGN(1,FJAM,8)
        READ(1)CASE,IS,DBTAIL,NRDN,NSAM,MP,(Q(I),I=1,MP)
CCC     Impulse response read into Q array but not used.  Q shared with SMO.
        IAVE=MAX0(IAVE,IS/LAPP)          ! Average at least over 1/B.
        IAVE=MIN0(IAVE,50)               ! Raise OUT(50) for more room.'
        FINBLN=INBLEN
        FIS=IS
        S=TWOPI*DOPP/FIS                 ! Used as 'per sample'.
        ROT=CMPLX(COS(S),SIN(S))
        TYPE 411,DOPP,ROT
411     FORMAT(1X8F12.6)
        TYPE 41,RUN(1),FJAM(1),FREF(1),NT,NBS0,KQ,LAPP,IAVE,INBLEN,INBDOP
     .,POW,TAU,GAIN,DOPP                 ! Echo to verify.
        IF(MP.GT.303)STOP        'SIMI:  Q ARRAY TOO SMALL'
        DO 1 I=1,NSC             ! Add IS*NBS0 to all ND(I).
1       ND(I)=IS*NBS0+ND(I)
        WRITE(3,42)NSC,(ND(I),CR(I),I=1,NSC)     ! Note scatterer data.
        WRITE(3,44)IS,DBTAIL,NRDN,TAU,GAIN,NT,KQ,LAPP,IAVE,DOPP,INBDOP
44      FORMAT(/' IS='I2,3X'DBTAIL= 'F4.0,3X'NRDN='I5,3X
     .'TAU='F8.0,3X'GAIN='F5.0,3X'NT='I4,3X'KQ='I2,
     ./' LAPP='I2,2X' IAVE='I4,4X'DOPP='F9.6,4X'INBDOP='I6)
        MORE=.TRUE.
        MAXDEL=0
        DO 4 I=1,NSC             ! Find max scatterer delay.
4       MAXDEL=MAX0(MAXDEL,ND(I))
        KS0=MAX0(MAXDEL,KQ*(NT-1))+1
        KS=KS0          ! Leave enough "past" to serve all scatterers.
        IF(KS.GT.501)STOP        'SIMI: KS too big.  Chge shift after stmnt 8.'
CCC     KS points to current H sample.
        SAMPS=0.                 ! Here to signal phase 1 skip
        IF(POW.GT.0.)GO TO 14    ! Skip first phase.
        ASSIGN 13 TO NOMORE
        ASSIGN 3 TO INFULL
        M=1
        R2=0.                    ! Initialize for zero crossing count.
        S2=0.
        CROSS=0.
        POW=0.
        YY=0.                    ! Will contain !Y!**2.
        SMOCNT=0.                ! For SMO averages.
        DO 30 I=1,2664           !**! Zero moment matrix.
30      BOTH(I)=0.
        GO TO 12                 ! for JAM input, phase 1
3       ASSIGN 6 TO INFULL
5       IF(KS.GT.KLIM)GO TO 8            ! Seek more jammer output.
6       CC=(0.,0.)
```

```
        DO 7 I=1,NSC
 7      CC=CC+CR(I)*H(KS-ND(I))
        S=REAL(CC)**2+AIMAG(CC)**2       !  !Y!**2
        IF(MOD(KS,IS).NE.0)GO TO 29      ! Feed SMO at intervals of 1/B.
        DO 27 I=1,NT                 ! Accumulate moments EY*.
 27     SMO(I,NT+1)=SMO(I,NT+1)+H(KS-K0*(I-1))*CONJG(CC)
        DO 28 I=1,NT                 ! Accumulate matrix of second moments.
        DO 28 J=1,NT
 28     SMO(I,J)=SMO(I,J)+H(KS-K0*(I-1))*CONJG(H(KS-K0*(J-1)))
        YY=YY+S
        SMOCNT=SMOCNT+1.
 29     POW=POW+S
        S1=S2              ! Count zero crossings for bandwidth estimate.
        R1=R2
        R2=SIGN(.125,REAL(H(KS)))
        S2=SIGN(.125,AIMAG(H(KS)))
        CROSS=CROSS+ABS(R2-R1)+ABS(S2-S1)
        SAMPS=SAMPS+1.
        KS=KS+1
        IF(SAMPS/FIS.GE.FINBLN)GO TO 13 ! Quit after INBLEN*IS samples.
        GO TO 5                      ! next sample, phase 1
 8      IF(.NOT.MORE)GO TO NOMORE,(13,22)       ! Stop if reach end of file.
        DO 9 I=1,500
 9      H(I)=H(I+500)                ! Shift last 500.
        M=11
        KS=KS-500                    ! Move pointer back 500 too.
 12     DO 10 I=M,20                 ! Fill H if possible.
        K=50*I-50
 10     READ(1,END=11)(H(K+J),J=1,50)
        KLIM=1000
        GO TO INFULL,(3,6,21)
 11     MORE=.FALSE.
        KLIM=K             ! KLIM is last cell filled.
        GO TO INFULL,(3,6,21)
 13     POW=POW/SAMPS
        REWIND 1           ! Back to start of jam file.
        READ(1)            ! Skip first record.
        MORE=.TRUE.
        WRITE(3, 45)SAMPS,POW
        TYPE 45,SAMPS,POW
 45     FORMAT(/F8.0,' samples, average power='F12.6)
 CCC    Calculate best possible cancellation ratio.
        DO 34 I=1,2664   !**!
 34     BOTH(I)=BOTH(I)/SMOCNT
        S=YY/SMOCNT                  ! Calc !Y!**2 average.
        SCA=1./SQRT(S)               ! Scale to unit residue power.
        CALL MATV(SMO,NT,NDIM)
        DO 32 I=1,NT
        CC=(0.,0.)                   ! Scratch
        DO 33 J=1,NT
 33     CC=CC+SMO(I,J)*SMO(J,NT+1)
 32     WST(I)=CC*SCA                ! "Theoretic weights"
```

```
      WRITE(3,413)(WST(I),I=1,NT)
413   FORMAT(/' "Theoretical Weights"'//(1X6F12.4))
      CC=(0.,0.)                ! Calc UU=(E*Y).inv(M).(EY*) average.
      DO 31 I=1,NT
31    CC=CC+CONJG(SMO(I,NT+1))*WST(I)
      CAN=1.-CC*SCA             ! Real cancellation ratio.
      OPT=10.*ALOG10(CAN)
      S=1./SCA
      S1=REAL(CC)
      TYPE 412,S1,S,CAN,OPT,SMOCNT
      WRITE(3,412)S1,S,CAN,OPT,SMOCNT
412   FORMAT(/4X'!UU!**2'5X'!Y!**2'8X'CR'8X'CR(dB)'3X'SMOCNT'
     .//1X2G12.4,F11.5,F9.1,F9.0)
14    SCA=1./SQRT(POW)
      DO 15 I=1,NSC             ! Scale REF so power is 1.
15    CR(I)=CR(I)*SCA
CCC   Initialize for simulation.
      DO 16 I=1,NT
16    WST(I)=(0.,0.)
      ASSIGN 22 TO NOMORE
      ASSIGN 21 TO INFULL
      KO=1                      ! For OUTs
      KR=0                      ! For RES(dB)
      IT=0                      ! Iteration count (multiples of IAVE)
      TURN=ROT                  ! Initialize for Doppler shifting.
      KS=KS0
      M=1
      GO TO 12                  ! for JAM input, phase 2
CCC   Calculate current back-scatter.
21    BE=(0.,0.)                ! "Beam"
      DO 17 I=1,NSC
17    BE=BE+CR(I)*H(KS-ND(I))
      IF(IT/IS.LT.INBDOP/LAPP)GO TO 26   ! Dopp after INBDOP 1/B steps.
      TURN=TURN*ROT             ! Cumulative phase shift
      BE=BE*TURN                ! Doppler shift beam.
CCC   Calculate current system output.
26    CC=BE                     ! Beam weight fixed at 1.
      DO 18 I=1,NT
18    CC=CC-WST(I)*H(KS-KO*(I-1))
CCC   Save magnitudes for progress report.
      OUT(KO)=REAL(CC)**2+AIMAG(CC)**2
25    KO=KO+1
      IF(KO.LE.IAVE)GO TO 19    ! See if OUT bin full.
      S=0.                      ! Average last IAVE residues.
      DO 24 I=1,IAVE
24    S=S+OUT(I)
      KR=KR+1
      IF(KR.GT.888)STOP'SIMI: RES(888) used up; raise 888 or IAVE.'!**!
      RES(KR)=10.*ALOG10(S/IAVE)      ! and current residue.
      TYPE 46,IT,RES(KR)        ! Note progress on CRT.
```

```
46      FORMAT(I6,F8.1)
        IT=IT+IAVE
        IF(IT/IS.GE.INBLEN/LAPP)GO TO 22        ! Stop after INBLEN 1/B steps.
        KQ=1
19      DO 20 I=1,NT              ! Calc current (U-star)*Beam
20      SMIN(I)=CC*CONJG(H(KS-KQ*(I-1)))
        DO 23 I=1,NT              ! Update Weight-stars.
23      WST(I)=DECAY*WST(I)+DEL*SMIN(I)
        KS=KS+LAPP                          !Increment time..
        IF(KS.GT.KLIM)GO TO 8     ! Need more jammer output.
        GO TO 21                  ! for  next sample step
22      IF(SAMPS.GT.0.)CRBW=CROSS/SAMPS*FIS     ! Cycles per 1/B !
        TYPE 47,CRBW,IT,(WST(I),I=1,NT)
        WRITE(3, 47)CRBW,IT,(WST(I),I=1,NT)
47      FORMAT(/' CRBW='F8.4,
     .' Cycles per 1/B'I12,' Iterations'8X'Final WST:'/(6F12.4))
        CALL CLOSE(1)
        CALL INPLOT(2,RUN(1))             ! Initialize plotter.
        WRITE(2,49)3,3                    ! Axis descr.
49      FORMAT(' PU 4500 -1200LBINTERVALS of 1/B'A1,'PU-1200 4450 DR0,1;
     .LBCANCELLATION (dB)'A1,'DR;')
        WRITE(2,48)HEAD,RUN,3             ! Head plot.
48      FORMAT(' PU0 10700LB'8A4,16X9A4)
        WRITE(2,414)OPT,3
414     FORMAT(' PU 7600 9099 LBOptimum='F6.1,' dB'A1)
        SAM(2)=FLOAT(LAPP)*FLOAT(IT)/.IS        ! Number of intervals of 1/B
        CALL PLU73(KR,SAM,RES,-40.,0.,0.,1.,1,-1,';',0,';',1,2)
        CALL CLOSE(2)
        KR0=MAX0(1,KR-95)         ! Print last 96 residue averages.
        WRITE(3,410)(RES(I),I=KR0,KR)   ! Change KR0 to 1 if want 'em all.
410     FORMAT(/1X12F6.1)
        CALL CLOSE(3)
        GO TO 2          ! for next case, if any
        END
```

```
      FUNCTION GAMMA(X)
CCC   Uses recurrence foll by Hastings poly approx (Abrams & Segun, p. 257)
CCC   Acc 6 or 7 dec for 1<X<2, declining twd vert asymptotes.
CCC   Can overflow for X near vert aymptote.  No check.
      REAL B(9)        ! Compiler rounds B(I).
      DATA B/.035868343,-.193527818,.482199394,-.756704078,.91820 357
     .,-.897056937,.988205891,-.577191652,1./
      Y=X
      FAC=1.
4     IF(Y)1,2,3
1     FAC=FAC/Y        ! Recurrence for neg arg.
      Y=Y+1.
      GO TO 4
5     Y=Y-1.           ! Recurrence for arg > 1.
      FAC=FAC*Y
3     IF(Y.GT.1.)GO TO 5
      S=B(1)
      DO 6 I=2,9       ! Series is for GAMMA(Y+1).
6     S=S*Y+B(I)
      GAMMA=S*FAC/Y
      RETURN
2     TYPE 40,X
40    FORMAT(' GAMMA FUNCTION:          X='F12.6)
      STOP    'GAMMA'
      END
```

```
      SUBROUTINE MATV(A,N,NDIM)
      COMPLEX A(NDIM,NDIM)
      DO 11 N1=1,N
      DO 12 J=1,N
      IF(J.EQ.N1) GO TO 12
      A(N1,J)=A(N1.J)/A(N1,N1)
12    CONTINUE
      DO 15 I=1,N
      IF(I.EQ.N1) GO TO 15
      DO 16 J=1,N
      IF(J.EQ.N1) GO TO 16
      A(I,J)=A(I,J)-A(I,N1)*A(N1,J)
16    CONTINUE
      A(I,N1)=-A(I,N1)/A(N1,N1)
15    CONTINUE
      A(N1,N1)=1./A(N1,N1)
11    CONTINUE
      RETURN
      END
```

```
      PROGRAM CKJAM
      REAL CASE(2),Q(303),COR(606),WORD(2)
      DATA CASE(2),WORD(2)/'.JAM','.COR'/
1     TYPE 40
40    FORMAT(' Enter JAM file name as 1XA4.')
      ACCEPT 4,CASE(1)
      CALL ASSIGN(1,CASE,8,'OLD')
      READ(1)WORD(1),IS,DBTAIL,NRDN,NSAM,MP,(Q(MP+1-I),I=1,MP)
      TYPE 4,WORD(1),IS,DBTAIL,NRDN,NSAM,MP,(Q(I),I=1,MP)
4     FORMAT(1XA4,I3,F6.0,3I6/(6F12.6))
      IF(MP.GT.303)STOP        'CKJAM:  MP TOO BIG.  UP Q, COR DIMENSIONS.
      CALL CLOSE(1)
      CALL ASSIGN(1,WORD,8,'NEW')
      WRITE (1,4)WORD(1),IS,DBTAIL,NRDN,NSAM,MP,(Q(I),I=1,MP)
      DO 3 I=1,MP
      ID=I-1
      S=0.
      DO 2 J=1,MP-ID
2     S=S+Q(J)*Q(J+ID)
      IF(I.EQ.1)T=S
3     COR(I)=S/T
      J=MIN0(MP,32)    ! Change to reduce TTY list size.
      WRITE (1,46)(COR(I),I=2,J)
46    FORMAT(/(8F10.6))
      CALL CLOSE(1)
      GO TO 1
      END
```

```
      SUBROUTINE GL73(Q,N,F,DE,EN1,EN2,VS,VG)
      DIMENSION Q(N)
      IF(VS.LE.VG)GOTO8
      S=VG
      G=VS
      GOTO7
8     G=Q(1)
      S=Q(1)
      DO 1  I=2,N
      G=AMAX1(Q(I),G)
1     S=AMIN1(Q(I),S)
      IF(VS.GE.VG)GOTO7
      S=AMAX1(S,VS)
      G=AMIN1(G,VG)
C     PICKS ROUND NUMBER PLOT BOUNDARIES & AXIS POINTS TO MARK.
C     PLOT RANGE WILL BE FROM DE*EN1*F TO DE*EN2*F, WHERE
C     F IS A POWER OF 10, EN1 AND EN2 ARE INTEGERS DIFFERING
C     BY AT MOST 10, AND DE IS .2  .5  OR  1.
7     D=G-S
      IF(D.GT.0.)GOTO2
      IF(D.LT.0.)STOP 'GL73:  NO OVERLAP OF LIMITS AND PLOT DATA'
      IF(D.EQ.0.)STOP 'GL73:  CANNOT SCALE.  MAX=MIN.'
2     T=ALOG10(D)
      U=AINT(T)
      IF(U.EQ.T)GOTO3
      IF(D.GE.1.0)GOTO3
      U=U-1.0
3     F=10.0**U
4     GSP=D/F                          ! SCALE RANGE TO 1, 10.
      GP=G/F
      SP=S/F                           ! F IS PWR OF 10 SCALE FACTOR.
      DE=1.0                           ! NORMALIZED LABEL INTERVAL
      IF(GSP.LT.5.0)DE=.5              !  1.0   .5   OR   .2
      IF(GSP.LT.2.0)DE=.2
      T=AMAX1(ABS(GP),ABS(SP))
      EN2=AINT(GP/DE)
      IF(GP.LT.0.)GOTO5
      IF(ABS((DE*EN2-GP)/T).GE..0001)EN2=EN2+1.      ! FUZZY COMPARE
5     GPP=EN2*DE                                     ! MAX LABEL
      EN1=AINT(SP/DE)
      IF(SP.GE.0.)GOTO6
      IF(ABS((DE*EN1-SP)/T).GE..0001)EN1=EN1-1.
6     SPP=DE*EN1                       ! MIN LABEL
      RETURN
      END
```

```
CCC             27-SEP-83
C       Plots curve of NPTS points described by EXIZ, WISE arrays.
C       WISE has NPTS values; EXIZ has 2 values: least and greatest.
C       IF(HASW.EQ.0.) USES UP TO 11 AXIS LABELS;
C       IF(HASW.NE.0.), USES EVERY OTHER IF MORE THAN 8.
C       PARITY OF ENS DETERMINES IF TOP OR BOTTOM LABEL USED WHEN SKIPPING.
C       EXPECTS PLOTTER'S P1,P2 IN IP73(4).
C       NEWAX=0: ASSUMES SCALING DONE; ONLY PLOTS.
C       NEWAX=1: SCALE, PLOT, LABEL AND TICK AXES.
C       NEWAX=2: SCALE AND PLOT.  OMIT AXIS STUFF.
C       NEWAX=3: ASSUMES SCALING DONE.  PLOTS CURVE.  DOES AXIS STUFF.
C       NEWAX=4: SCALE ONLY (USEFUL FOR FORCING SCALES).
C       LINE IN RANGE -1 TO 6 PICKS PATTERN (HP PAGES 4-6,7).
C       IF VS < VG, LIMITS PLOT TO VS < WISE < VG.
C       IF VS > VG, FORCES PLOT RANGE TO INCLUDE VG TO VS.
C       IF PLCHR IS A PRINTING CHARACTER, IT WILL BE USED FOR A SYMBOL
C       PLOT. ELSE GET LINE PLOT (SUGGEST ;).  SEE HP PAGE 4-5.
C       TO GET SYMBOL PLOT WITHOUT CONNECTING LINES, USE LINE= 0
C       IF NLAB>0, LABELS WI 'CHLAB' EVERY NLAB POINTS, STARTING ABT NLAB/2.
C       IF IPEN=2, USES PEN 2; ELSE PEN 1.
C       IF ITIX=4, TICKS ALL 4 SIDES; ELSE TICKS ONLY LEFT AND BOTTOM.
        SUBROUTINE PLU73(NPTS,EXIZ,WISE,VS,VG,ENS,HASW,NEWAX,LINE,
     .PLCHR,NLAB,CHLAB,JPEN,JTIX)
        DIMENSION EXIZ(2),WISE(NPTS)
        REAL EN1(2),EN2(2),CX(2),CY(2),F(2),DE(2),A(2),B(2),ENZ(2)
        COMMON /Q73/IP73(4)
        DATA CX,CY/-4.,-9.,-2.,-.25/
        DATA IP73/1250,1250,9250,7000/
        IPEN=2
        IF(JPEN.NE.2)IPEN=1
        ITIX=4
        IF(JTIX.NE.4)ITIX=2
104     WRITE(2,405)IPEN,IP73   !INITIALIZE AND SET SCALE.
405     FORMAT(' IN;SP'I2,'IP'4I6,'SC0,9999,0,9999;R;DR;')
        JMP=NEWAX+1
        GOTO(108,113,113,108,113),JMP
113     WRITE(2,404)                    ! DRAW BOX.
404     FORMAT(' PU0,0PD0,9999,9999,9999,9999,0,0,0PU')
        CALL GL73(EXIZ,2,F(1),DE(1),EN1(1),EN2(1),0.,0.)        ! GET X DIVS.
        CALL GL73(WISE,NPTS,F(2),DE(2),EN1(2),EN2(2),VS,VG)     ! GET Y DIVS.
        DO 117 I=1,2    ! SCALE X,Y TO PLOT VARIABLES.
        ENZ(I)=9999./(EN2(I)-EN1(I))
        A(I)=ENZ(I)/(F(I)*DE(I))
117     B(I)=EN1(I)*ENZ(I)
        EX(V)=V*A(1)-B(1)       ! SCALING FUNCTIONS
        EY(V)=V*A(2)-B(2)
        EKS(I)=FLOAT(I-1)/FLOAT(NPTS-1)*(EXIZ(2)-EXIZ(1))+EXIZ(1)
        IF(JMP.GE.5)GOTO107
108     WRITE(2,416)IP73,LINE,PLCHR     ! BOUND; SET LINE PATTERN, PLOT MODE.
416     FORMAT(' IW'4I6,'LT'I4,'SM'A1)
CCCC    IN SYMBOL MODE, USER RESPONSIBLE FOR DENSITY OF PLOTTED SYMBOLS.
        WRITE(2,411)((EX(EKS(I)),EY(WISE(I))),I=1,NPTS) ! PLOT CURVE.
411     FORMAT(' PU'2F8.1,'PD'(1X2F8.1))
        IF(NLAB.LE.0)GOTO111    ! LABEL CURVE WITH CHLAB IF NLAB>0.
        DO 109 I=1,NPTS
109     IF(MOD(I+NLAB/2,NLAB).EQ.0)WRITE(2,412)EX(EKS(I)),EY(WISE(I))+
     .140.,CHLAB,3
```

```
412     FORMAT(' PU'2F8.1,'LB'2A1)
111     WRITE(2,414)     ! RESTORE LINE PLOTTING AND OPEN WINDOW.
414     FORMAT(' PU;SM;IW')
        GOTO(107,112,107,112,107),JMP
CCCC    TICK AND LABEL AXES.
112     LN=2
        LL=1
        XX=0.
        YY=0.
        ZZ=100.
        DO 100 L4=1,ITIX
        L=2-MOD(L4,2)
        N=EN2(L)-EN1(L)
        EN=EN1(L)
102     WRITE(2,406)0,LN          ! SET BIG TICKS.
406     FORMAT(' TL'2I3)
        GOTO(105,106),L
105     WRITE(2,407)(EN-EN1(1))*ENZ(1),YY        ! BIG X TICK.
407     FORMAT(' PU'2F8.1,'XT')
        GO TO 101
106     WRITE(2,409)XX,(EN-EN1(2))*ENZ(2)        ! BIG Y TICK.
409     FORMAT(' PU'2F8.1,'YT')
101     DT=.1
        NLIL=9
        IF(HASW.EQ.0..OR.N.LE.7)GOTO110
        DT=.2
        NLIL=4
        IF(AMOD(EN-EN2(L)-ENS,2.).NE.0.)GOTO103
C       OTHERWISE LABEL BIG TICK.
110     IF(L4.LE.2)WRITE(2,408)CX(L),CY(L),EN*DE(L)*F(L),3
408     FORMAT(' CP'2F6.2,';LB'G11.4,A1)
103     IF(EN.GE.EN2(L))GOTO118
        WRITE(2,406)0,LL          ! SET LIL TIX.
        GOTO(114,115),L
114     WRITE(2,407)(((I*DT+EN-EN1(1))*ENZ(1),YY),I=1,NLIL)      ! LIL TIX.
        IF(NLIL.EQ.9)WRITE(2,407)(5.*DT+EN-EN1(1))*ENZ(1),YY-ZZ ! LENGTHEN.
        GO TO 116
115     WRITE(2,409)((XX,(I*DT+EN-EN1(2))*ENZ(2)),I=1,NLIL)      ! LIL TICKS.
        IF(NLIL.EQ.9)WRITE(2,409)XX-ZZ,(5.*DT+EN-EN1(2))*ENZ(2) ! LENGTHEN.
116     EN=EN+1.
        GOTO102
118     IF(L4.NE.2)GO TO 119
        YY=9999.
        LN=-LN
        LL=-LL
        ZZ=-ZZ
119     IF(L4.NE.3)GO TO 100
        YY=0.
        XX=9999.
100     CONTINUE
107     WRITE(2,494)     ! FLUSH BUFFER TO FINISH EACH CURVE.
494     FORMAT(128X/128X)
        RETURN
        END
```

```
CCC     ASSIGNS LOGICAL UNIT LUN FOR PLOTS TO DEVICE TTN:
C       AND INITIALIZES PLOTTER, WHICH IS CONNECTED TO TTN:
C       EXAMPLE:        CALL INPLOT(2,'TT2:')
C       WORTHIE ROUTINES WDB AND WSOP USE LUN=2
        SUBROUTINE INPLOT(LUN,DEV)
        CALL ASSIGN(LUN,DEV,4)
        WRITE(LUN,4)27,27
4       FORMAT(1H+A1,'.I40;;17:'A1,'.N;19:')
        RETURN
        END
```

APPENDIX B

SIMULATION OF COLORED NOISE

APPENDIX B

SIMULATION OF COLORED NOISE

The jammer signal is represented by a set of consecutive samples, with the spacing between samples specified as an input to the program. Each sample of the jamming is a complex number, selected from a distribution with independent zero-mean Gaussian quadrature components. The correlation between consecutive samples depends on sample spacing and the frequency spectrum.

The effect of jammer frequency spectrum on the steady state performance of a scatter canceller has been discussed in earlier reports. When the frequency spectrum is strictly limited to a bandwidth B, a tap spacing of 1/B in the canceller can provide arbitrarily good cancellation, provided a sufficient number of taps is used. With a rectangular spectrum, i.e., constant spectral density over the bandwidth B, consecutive samples spaced by 1/B are independent. However, it was shown that a large number of taps extending beyond the scatter interval are required with this spectrum. In most cases of interest, the jammer bandwidth is wider than the receiver pass band. The spectrum at the input to the adaptive processor is then determined by the frequency response of the receiver. The rectangular spectrum, with constant amplitude response over the pass band and zero response outside the passband is difficult to approximate closely with a filter. More representative filters have a varying response over the band.

A Gaussian frequency spectrum is sometimes assumed for convenience of analysis. It was shown in earlier reports that a high sampling rate

is required to achieve good scatter cancellation when the canceller input has a Gaussian spectrum. A sampling interval of roughly 1/3B is required.

A more representative spectral shape is the cosine spectrum,

$$\cos\left[\frac{\pi(f - fo)}{B}\right] \text{ for } |f - fo| < B/2$$

and zero for frequencies outside this band. It was shown to be a suitable spectrum for use in a scatter canceller. This spectrum is strictly band-limited and does not require an excessive number of taps beyond the scatter interval to achieve 30 dB or so of cancellation. With wide bandwidth jamming, this spectrum is determined by the receiver transfer function and can be selected to facilitate scatter cancellation.

There are two well known methods of generating a sequence of random samples with a specified frequency spectrum. One method is to generate a set of random samples representing the spectral components and then Fourier transform (FFT) these samples into a corresponding set of time samples. This algorithm generates a finite block of data, with the number of correlated samples limited by the FFT dimensionality. The second method is to convolve a sequence of independent samples with the appropriate impulse response function for the desired spectrum. The latter method is used in the simulation program.

The cosine spectrum is strictly band-limited and, at base band, has the form

$$S(f) = \frac{\pi}{2B} \cos\left(\frac{\pi f}{B}\right) \qquad |f| \le B/2$$

$$= 0 \qquad \text{otherwise} \tag{1}$$

One impulse response function which generates this spectrum is[1]

$$Q(t) = \int_{-\infty}^{\infty} \sqrt{S(f)} \cos 2\pi ft \, df$$

$$= \frac{\pi}{B} \int_0^{B/2} \sqrt{\cos\left(\frac{\pi f}{B}\right)} \cos 2\pi ft \, df \tag{2}$$

$$= \frac{\Gamma\left(\frac{3}{2}\right) \Gamma\left(Bt - \frac{1}{4}\right)}{2^{3/2} \Gamma(Bt + 5/4)} \sin\left[\pi\left(Bt - \frac{1}{4}\right)\right]$$

Let $\{u_n\}$ denote a set of zero-mean independent Gaussian samples with $E|u_n|^2 = 1$. The set of jammer samples $\{v_n\}$ with the frequency spectrum $S(f)$ is obtained by convolving the impulse response function with the sequence u_n,

$$v_n = \sum_{m=-\infty}^{\infty} Q_{|m|} u_{n-m} \tag{3}$$

[1] Bierens deHaan, "Nouvelles Tables d'Intégrales Définies", 1867, Hafner Publishing Co., Table 41.

In (3), the Q_m are samples of Q(t), given by

$$Q_m = \frac{\Gamma\left(\frac{m}{I_s} - \frac{1}{4}\right) \sin\pi\left(\frac{m}{I_s} - \frac{1}{4}\right)}{\Gamma\left(\frac{m}{I_s} + 5/4\right)} \quad , \quad m = 0(1)\infty$$

The constant factor in Q_m is dropped since the $\{Q_m\}$ are normalized later to maintain the same average power in the $\{u_n\}$ and $\{v_n\}$. The number of samples per time interval 1/B is denoted by I_s.

The required sequence of independent samples u_n is obtained by using a random number generator and

$$u_n = \sqrt{-\log(z_1)} \exp\{2\pi i\, z_2\} \quad , \qquad (5)$$

where z_1 and z_2 are independent random variables uniformly distributed in the interval (0, 1).

The sum in Eq. (4) is truncated to

$$v_n = \sum_{m=-M}^{M} Q_{|m|}\, U_{n-m} \quad . \qquad (6)$$

An easy and reasonable truncation criterion is to require that, on the average, the power in the cut-off tail is some preassigned fraction of the power in the truncated sum used. For the symmetrical impulse response, the average output power from the truncated series is

$$T_M = Q_o^2 + 2 \sum_{m=1}^{M} Q_m^2 \quad . \tag{7}$$

The power in the tails when the convolution is truncated at M is

$$S_M = 2 \sum_{m=M+1}^{\infty} Q_m^2 \quad . \tag{8}$$

A conservative bound on the power in S_M is obtained by replacing the $\overline{\sin^2}$ term by unity. Also, the ratio of gamma functions is replaced by an asymptotic expression, giving

$$S_M \leq 2 \sum_{m=M+1}^{\infty} \frac{\Gamma^2\left(\frac{m}{I_s} - \frac{1}{4}\right)}{\Gamma^2\left(\frac{m}{I_s} + \frac{5}{4}\right)} = 2 \sum_{m=M+1}^{\infty} \left(\frac{m}{i}\right)^{-3} (1+\varepsilon_m)$$

$$< 2 (1 + \varepsilon_M) \sum_{m=M+1}^{\infty} \left(\frac{m}{I_s}\right)^{-3} \tag{9}$$

$$< 2 (1 + \varepsilon_M) \int_M^{\infty} i^3 \frac{dm}{m^3} = (1 + \varepsilon_m) \frac{I_s^3}{M^2}$$

where ε_m is the fractional error in the asymptotic expansion. Since ε_m will be small compared to unity for any values of M of interest, it is dropped.

The allowable fractional error due to truncation of the impulse response function (FERR) is specified as an input to the program. The impulse response series of (6) is truncated when

$$\frac{S_M}{T_M} \leq \text{FERR} \quad . \tag{10}$$

From (9), a conservative bound is

$$M > \sqrt{\frac{I_s^3}{(\text{FERR})\ T_M}} \quad . \tag{11}$$

This bound is used in the simulation. In the examples in this report, where FERR = 10^{-4} and $I_s = 4$, the value of M is 101. In these cases, the 2-sided impulse response function used in generating a colored noise jammer output contains 203 terms.